Die umfassende Software – Entwicklungsumgebung zur einfachen Anwendungsentwicklung mit Microsoft Visual FoxPro

VISUAL EXTEND 11.0



Deutsches Benutzerhandbuch

dFPUG c/o ISYS GmbH

Venelina Jordanova, Uwe Habermann

Copyright

1. Einleitung

von Rainer Becker

Herzlich Willkommen zur neuen Version 11.0 von Visual Extend, auf die wir ganz besonders stolz sind! Denn es ist das größte Update, welches für das bekannte Framework bisher erstellt wurde. Klar, dieser sich wiederholende Marketingspruch wird sowohl für Visual FoxPro als auch für Visual Extend langsam etwas langweilig - nichts desto trotz trifft diese Aussage aber erneut bei beiden Produkten für viele Bereiche zu! Aber fangen wir mit Visual FoxPro 9.0 an:

1.1. Basierend auf Visual FoxPro 9.0

Visual Extend 11.0 basiert auf Visual FoxPro 9.0, beide neue Versionen sind seit Anfang 2005 im Handel erhältlich. Abgesehen davon, dass Visual Extend 11.0 die Version Visual FoxPro 9.0 als Voraussetzung benötigt, gibt es aber viele weitere Gründe, sich die neueste Version von Visual FoxPro im Detail anzuschauen bzw. zu erwerben. Visual FoxPro 9.0 bietet Ihnen unter anderem:

- Wesentliche Erweiterungen im Bereich der Datenbankengine, insbesondere der SQL-Syntax sowie Aufhebung vieler der bisherigen Einschränkungen von Visual FoxPro.
- Viele Jahre lang insbesondere im deutschsprachigen Raum gefordert und ersehnt, erfolgte endlich die komplette Neuerstellung des Berichtsdesigners und eine grundsätzliche Überarbeitung der Berichtsausführung mit überzeugenden Ergebnissen.
- Diverse Verbesserungen in der Benutzeroberfläche wie Docking/Anchoring für Masken, verbesserte Grafikunterstützung, Autotext u.v.m. - und Format Z ist auch wieder zurück.

Aber auch viele Kleinigkeiten wurden bei der neuen Version bereinigt, verbessert und erweitert. Ein schöner Nachtrag ist übrigens eine kleine neue Eigenschaft für Grids:

Tipp: Rushmore-Optimierung in Grids
Eine neue Eigenschaft „Optimize“ steht für Grids zur Verfügung und stellt damit erstmals die lange vermisste Rushmore-Optimierung für die tabellarische Darstellung zur Verfügung. Jetzt ist das Grid nicht mehr langsamer als ein BROWSE-Befehl.

PS: Falls Sie also jemals in die Verlegenheit kamen, eine gefilterte Tabelle in einem Grid zu verwenden, setzen Sie diese Eigenschaft doch mal auf .T. (der Default ist natürlich .F.).

Die tatsächliche Liste der Verbesserungen wollen wir hier natürlich nicht komplett abdrucken, aber gehen Sie davon aus, dass die Endversion von Visual FoxPro 9.0 von der lange Zeit verfügbaren Public Beta erheblich abweicht und wesentlich umfangreicher geworden ist!

1.2. Die Kombination macht’s: All in One

Visual FoxPro 9.0 ist als objektorientierte Entwicklungsumgebung und als relationales Datenbanksystem in der neuen Version noch attraktiver für die Anwendungsentwicklung geworden. Das Framework Visual Extend nunmehr ergänzt das Werkzeug-Set von Visual FoxPro um die entscheidenden Komponenten zur schnellen Anwendungsentwicklung, oder neudeutsch „Rapid Application Development“, kurz RAD.

Dies geschieht zum einen durch die Bereitstellung eines umfangreichen Anwendungsrahmens mit vielen wichtigen Standardfunktionen für Ihre Anwendung wie die

- Verwaltung von Benutzern, Gruppen, Zugriffsrechten
- Datensicherung und -wiederherstellung
- Datenbankwartung und -reparatur
- Fehler-, Sperren-, User- und Änderungsprotokoll,
- Favoriten, Anpassen und Optionen, Infomaske
- Filtern, Berichtsausgabe incl. Ausgabe als PDF/Fax usw.

Mehr zu den vielen für Sie fertig vorbereiteten Funktionen der generierten Applikation lesen Sie im Kapitel „3.2. Eigenschaften von mit Visual Extend erstellten Anwendungen“.

Und dies geschieht zum anderen durch die Bereitstellung eines verhältnismässig kleinen Sets von Basisklassen, hauptsächlich in den Bereichen Formulare, Grids und Lookups in verschiedenen Geschmacksrichtungen. Und dazu die entsprechenden umfangreichen Builder, die wie ein Schweizer Multifunktionstaschenmesser zusammenwirken und die schnelle Konfiguration dieser Klassen durch den Entwickler erlauben.

Mehr zu den vielen für Sie fertig vorbereiteten Klassen und den dazugehörigen Buildern lesen Sie im Kapitel „3.3. Leistungsmerkmale für Entwickler“.

Ergänzt und abgerundet wird das Paket durch administrative Funktionen für Softwareentwickler sowie kleine und mittlere Softwarehäuser wie zum Beispiel

- Datenbank- und Anwendungsaktualisierung
- Aktivierungsschlüssel und Versionsupdate für Module
- Unterstützung von Fernadministration

Und dann wäre da noch unser neuer WebService für Ihre vereinfachte Registrierung von Visual Extend mitsamt Anforderung von Ersatzschlüsseln und… Doch wir wollen nicht das ganze Handbuch in der Einleitung vorwegnehmen. Lassen Sie uns nur den für Visual Extend besonders wichtigen Bereich der Builder kurz noch etwas genauer betrachten:

1.3. Noch produktiver durch neue Builder in Visual Extend 11.0!

Sofern Sie bereits mit Visual Extend arbeiten, werden Sie in fast jeder Zeile der nachfolgenden Auflistung der neuen Funktionen sofort erkennen, wie Ihnen dies die Alltagsarbeit erleichtern wird! Sofern Sie noch nicht mit Visual Extend arbeiten, erkennen Sie zumindest grob, wie umfangreich das aktuelle Update wirklich ist! Lesen Sie bitte:

- Sämtliche Eigenschaften des Applikationsobjektes sind im Application Wizard unter den erweiterten Optionen abrufbar – und später im Application Builder auch änderbar!
- In den Projekteigenschaften können Sie für sämtliche Builder die auswählbaren Klassen festlegen und auch gleich als Default sowie als AutoComplete definieren
- Die Project-Toolbox liefert Ihnen sämtliche projektspezifischen Klassen in Übersicht und zum direkten Drag&Drop oder (siehe rechte Maustaste) zum direkten Instanziieren.
- Der Project Documenting Wizard liefert Ihnen eine Schnittstelle zu einer speziellen VFX-Version von PDM zur Dokumentation Ihrer Anwendung
- Der Project Update Wizard erlaubt die halbautomatische statt manueller Aktualisierung bestehender Projekte auf neue Versionen und neue Builds von Visual Extend
- Der Dataenvironment-Builder (integriert mit Form-Wizard/Builder) erlaubt die visuelle Zusammenstellung des Dataenvironments incl. Integration des CA-Builders
- Sämtliche erweiterten Form-Builder haben Reiter für View-Parameter (mitsamt Eingabefeldern und Requery-Button), verlinkte Tabellen, benötigte Felder und zusätzliche Spalten für die Berichtsdarstellung
- Der Parent/Child-Builder erlaubt die visuelle Definition sämtlicher abhängiger Child-Masken statt die manuelle Definition in der onmore-Methode
- Im Language Setup Builder können Sie die Lokalisierung / Übersetzung der Benutzeroberfläche zur Laufzeit aktivieren, so dass Anwender selbst wählen können…
- In der Kundenliste können Sie nicht nur Aktivierungsschlüssel erzeugen, sondern auch gleich alle dazugehörigen Kundendaten verwalten.
- In der Updateverwaltung können Sie neue Versionen definieren und den Kunden gleich entsprechende Downloadrechte einräumen.
- In der Konfigurationsverwaltung können Sie nunmehr beliebig viele Definitionen hinterlegen, sämtliche VFX-Tabellen auf dem Backend-Server hinterlegen und eigene Spalten hinzudefinieren, die ebenfalls verschlüsselt abgespeichert werden.
- Der CursorAdaptor-Wizard erstellt Ihnen CursorAdaptor-Klassen automatisch für alle Tabellen in einem Datenbankcontainer in einer Bibliothek Ihrer Wahl
- Der AuditTrigger-Wizard erstellt Ihnen automatisch alle Trigger für den Audit-Trail für einzelne oder alle Tabellen eines Datenbankcontainers zwecks Nachverfolgung
- Im Systemobjekt können Sie über eine Definitionsmaske die Download-Skripte für Ghostscript, Acrobat Reader, OutlookYesNo sowie Update, Backup, DUN und DynDNS definieren und verwalten

- Platzieren Sie einen cDocumentManagement-Container auf einem leeren Reiter und definieren Sie die Dokumentenzuordnung zum aktuellen Datensatz mit dem Document Management Builder – und schon sind zentral alle Dokumentverweise in einer Tabelle.
- Platzieren Sie einen cBusinessGraph-Container auf einem leeren Reiter und – tja, der Builder ist leider doch noch nicht fertig <bg>.
- Platzieren Sie eine cComboPicklist auf Ihrer Editpage und verwenden Sie den ComboPickList-Builder für Definition und Festlegung der auswählbaren Werte. Und:
- Bearbeiten Sie die Werte in dem dazugehörigen Pflegeformular und verwenden Sie die Definition in der nächsten Maske erneut per Auswahl aus der Combobox-Übersicht!
- Oder verwenden Sie eine cTextCalculator, cTexteMail, cTextHyperlink, cLinkTextbox oder eine cTextTAPI-Klasse – dafür brauchen Sie nicht mal einen Builder…

Erwähnt haben wir für Sie NUR die neuen oder wesentlich erweiterten Builder bzw. Systemfunktionen aus dem VFX-Menü. Deshalb sagen wir:

Visual Extend 11.0 – Produktiver als je zuvor!

Und wir gehen davon aus, daß Sie uns bei dieser Aussage bedenkenlos zustimmen können!

2. Schnelleinstieg

2.1. Einführung

Visual Extend gehört seit vielen Jahren zu den leistungsfähigsten Zusatzprodukten von Visual FoxPro. Mit Visual Extend (im folgenden Text mit VFX abgekürzt) ist es möglich in wenigen Minuten den Rahmen für eine Visual FoxPro-Anwendung voll funktionsfähig zu erstellen. Wenn vor der Anwendungsentwicklung bereits eine Datenbank zur Verfügung steht, ist es ein Leichtes mit den Assistenten von VFX innerhalb kürzester Zeit Bearbeitungsformulare zu erstellen. Lernen wir die wichtigsten Eigenschaften von VFX kennen in dem wir die Arbeitsschritte zur Erstellung einer Anwendung durchgehen.

Zum Betrieb von Visual Extend 11.0 ist Visual FoxPro 9.0 erforderlich.

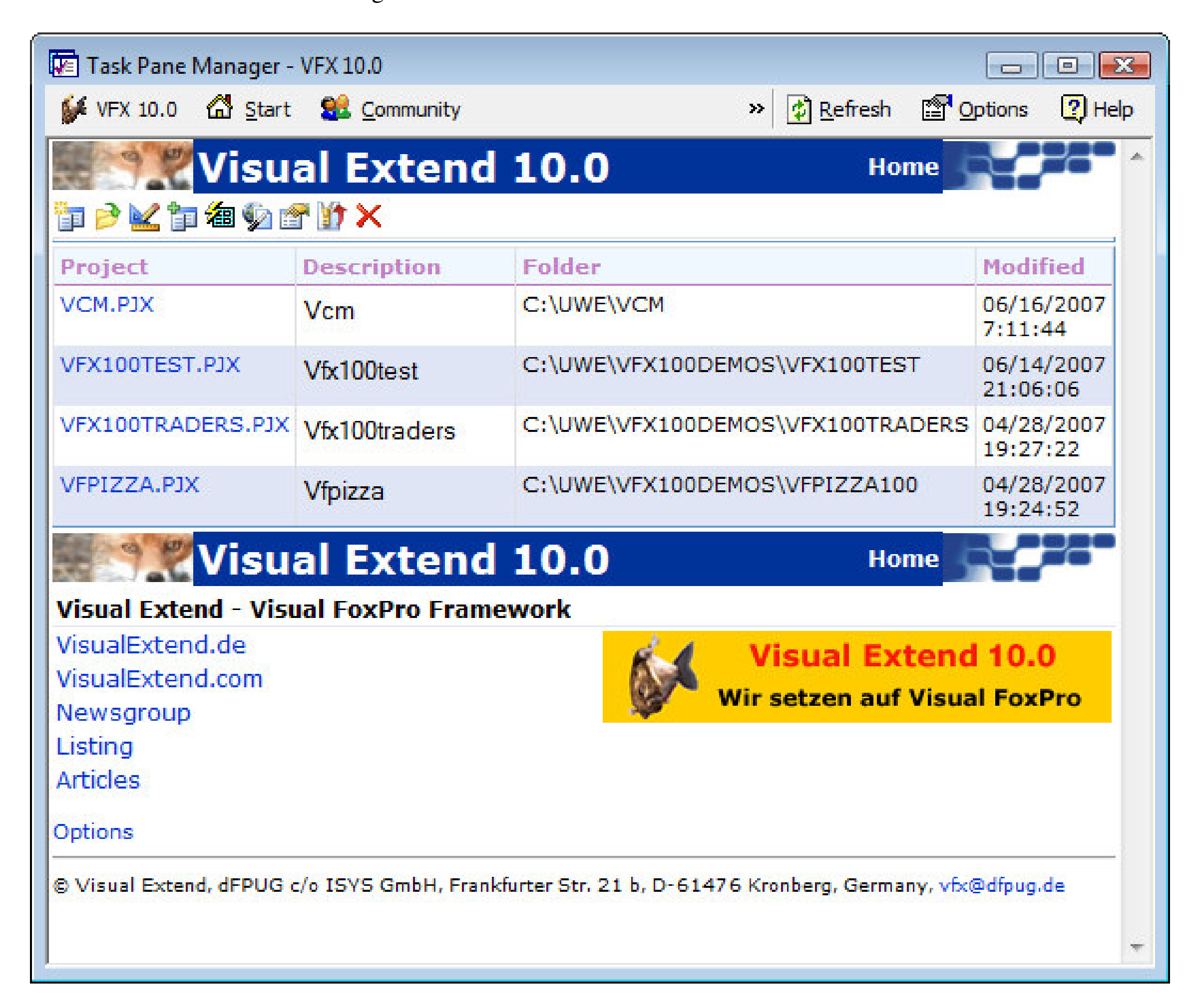
2.1.1. Installation

Nach der Installation von VFX ist es sinnvoll, das VFX-Menü in das Standardmenü von Visual FoxPro zu integrieren. Dazu ist in der Datei *Config.fpw* eine Zeile einzufügen:

```
Command = DO <VFX-Installationspfad>\builder\vfxmnu.app
```

2.1.2. VFX – Task Pane

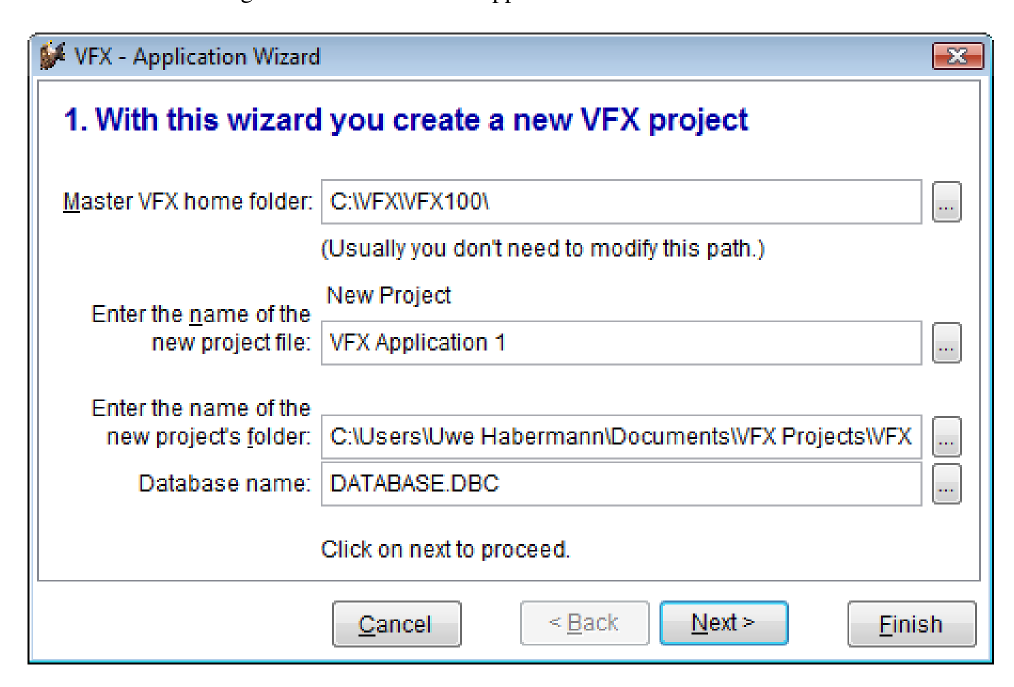
Beim ersten Start von VFP nach der Installation von VFX 11.0 wird automatisch die VFX 11.0 Task Pane in die Task Pane von Visual FoxPro integriert.



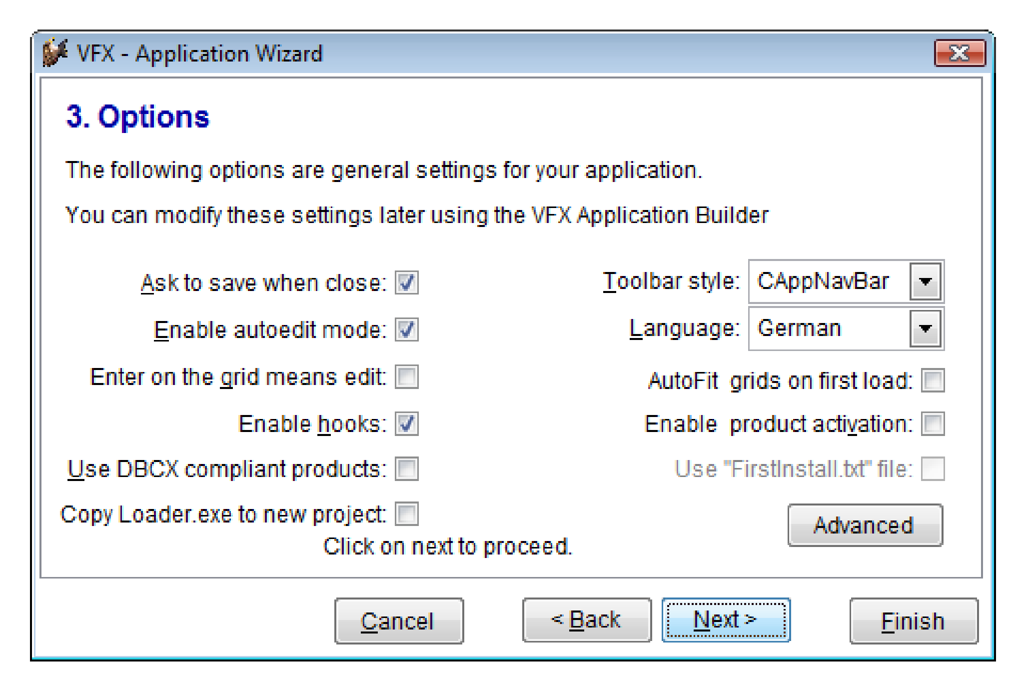
Ein nützliches Tool befindet sich in der VFX 11.0 Task Pane, der Application Manager. In einer Tabelle werden Informationen über alle VFX-Projekte verwaltet. Über den VFX – Application Manager kann ein Projekt geöffnet werden. Dabei wird automatisch der aktuelle Pfad auf den Projektordner gesetzt. Außerdem kann über den VFX – Application Manager ein „Rebuild all" durchgeführt werden. Dabei wird das Projekt komplett kompiliert. Änderungen in Include-Dateien werden dabei berücksichtigt.

2.1.3. VFX – Application Wizard

Eine neue Anwendung wird mit dem VFX – Application Wizard erstellt.



Beim ersten Aufruf des Wizard wird als Sprache für die zu erstellende Anwendung die Sprache der verwendeten FoxPro-Version vorgeschlagen. Bei jedem erneuten Aufruf wird die zuletzt verwendete Sprache vorgeschlagen. Nachdem die *Finish*-Schaltfläche gedrückt wird, werden aus der leeren VFX-Musteranwendung die Dateien in den neu erstellten Projektordner kopiert und anschließend kompiliert.



2.2. Funktionsumfang der neuen Anwendung

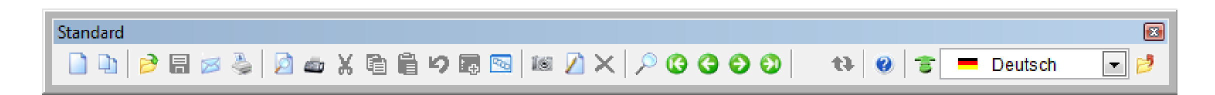
Die mit dem Application Wizard erstellte Anwendung kann sofort getestet werden. Dazu kann direkt aus dem Projekt-Manager das Hauptprogamm *Vfxmain.prg* gestartet werden. Wahlweise kann auch eine App- oder Exe-Datei erstellt und getestet werden. Dies ist während der Entwicklung normalerweise aber nicht erforderlich.

Die Anwendung startet mit einem Splashscreen. Als Bild für den Splashscreen wird eine Png-Datei verwendet, die der Entwickler leicht bearbeiten oder austauschen kann. Es ist möglich die Anzeige des Splashscreen zu unterdrücken. Nach Anzeige des Splashscreens baut sich der Hauptbildschirm auf und es erscheint der Anmel-debildschirm. Standardmäßig muss sich jeder Benutzer einer VFX-Anwendung mit einem Namen und einem Kennwort anmelden. Es ist möglich den Anmeldebildschirm zu umgehen und den Benutzer automatisch mit dem Windows-Anmeldenamen anzumelden.

2.2.1. Bedienung

Nach der Anmeldung wird die VFX-Anwendung ähnlich den Office-Anwendungen bedient. Benutzer, denen die Bedienung von Word oder Excel geläufig ist, können mit einer VFX-Anwendung praktisch sofort produktiv arbeiten.

2.2.2. Standard-Symbolleiste



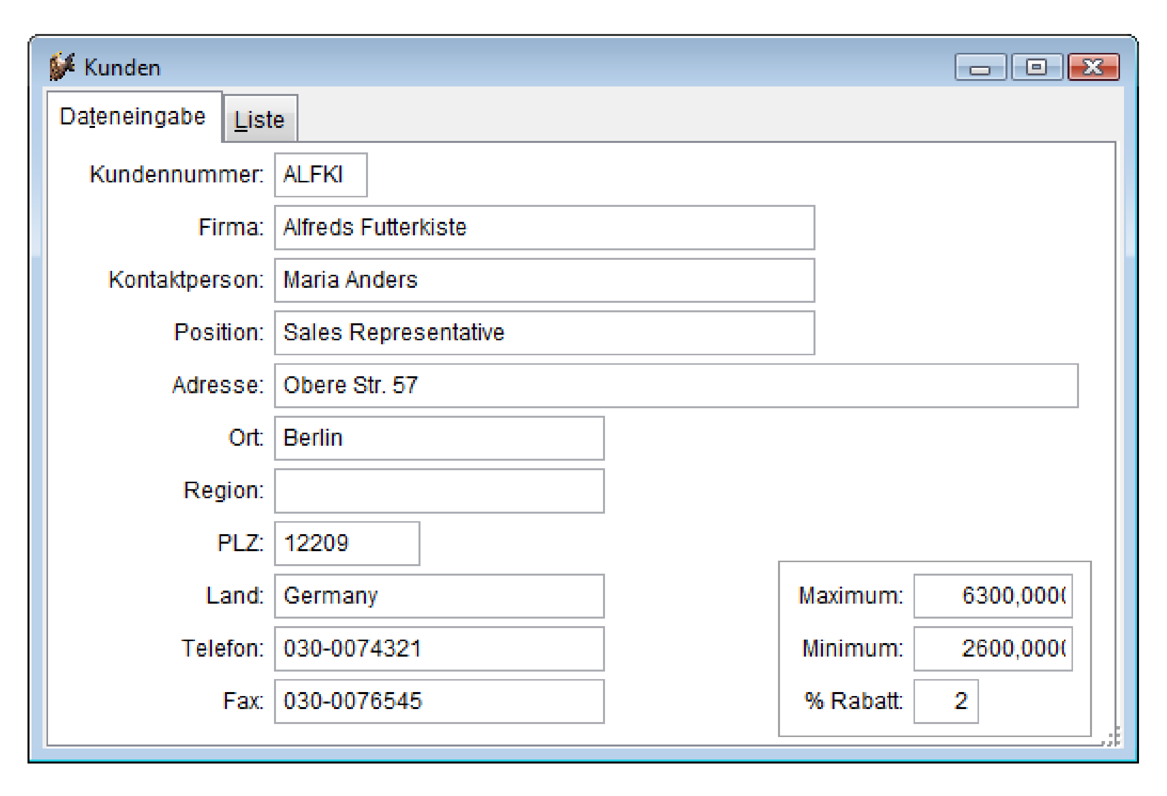
Viele der Schaltflächen der Symbolleiste sind in ihrer Funktion mit denen aus Office-Produkten identisch.

2.2.3. Öffnen-Dialog

Formulare werden standardmäßig über den Öffnen-Dialog gestartet. Der Öffnen-Dialog erscheint im Windows XP-Layout. Die Informationen der Formulare, die im Öffnen-Dialog angezeigt werden, stehen in der Tabelle *Vfxfopen.dbf*.



2.2.4. Formulare



Wenn für ein Formular die *lAutoedit*-Eigenschaft auf *wahr* eingestellt ist (das ist der Standardwert), sind ständig alle Steuerelemente auf dem Formular aktiviert. Der Anwender kann mit der Maus oder der Tastatur ein Steuerelement anwählen und sofort mit dem Bearbeiten der Daten beginnen. Das Formular wechselt automatisch in den Bearbeitungsmodus, sobald Daten interaktiv verändert werden.

Auf der Listenseite von VFX-Formularen befindet sich ein Grid. Standardmäßig kann in allen Spalten des Grid inkrementell gesucht werden. Dazu ist einfach der Fokus in die gewünschte Spalte zu setzen. Mit dem ersten Buchstaben- oder Zifferndruck wird die Sortierfolge auf diese Spalte umgestellt. Dabei wird bei Bedarf automatisch eine temporäre Indexdatei erstellt. Die Überschrift in der Spalte wird mit einem auf- oder absteigenden Pfeil, ähnlich der Darstellung im Windows-Explorer, gekennzeichnet.

Standardmäßig kann die Größe von VFX-Formularen vom Anwender zur Laufzeit geändert werden. Alle Steuerelemente werden dabei proportional in der Größe geändert. Innerhalb von Grids wird die Größe der Steuerelemente standardmäßig nicht verändert. Wenn ein Formular vergrößert wird, werden also mehr Zeilen und Spalten im Grid sichtbar.

Alle Einstellungen an Formularen werden benutzerspezifisch gespeichert. Wenn der Anwender das Formular erneut öffnet, erscheint das Formular an der Position des Bildschirms und in der Größe in der es zuletzt geschlossen wurde. Auch die Einstellungen der Grids (Spaltenbreiten, Spaltenfolge und Sortierung) werden gespeichert.

VFX-Formulare haben normalerweise eine private Datensitzung und können problemlos mehrfach geöffnet werden. Über eine Eigenschaft des Formulars (*lMultiinstance*) kann der mehrfache Aufruf verhindert werden.

2.2.5. Benutzerverwaltung

In VFX ist eine Benutzerverwaltung enthalten. Dazu gehören ein Formular zur Bearbeitung der Benutzerdaten, ein Formular zur Bearbeitung der Benutzerrechte, eine Verwaltung von Benutzergruppen sowie ein Anmeldebildschirm.

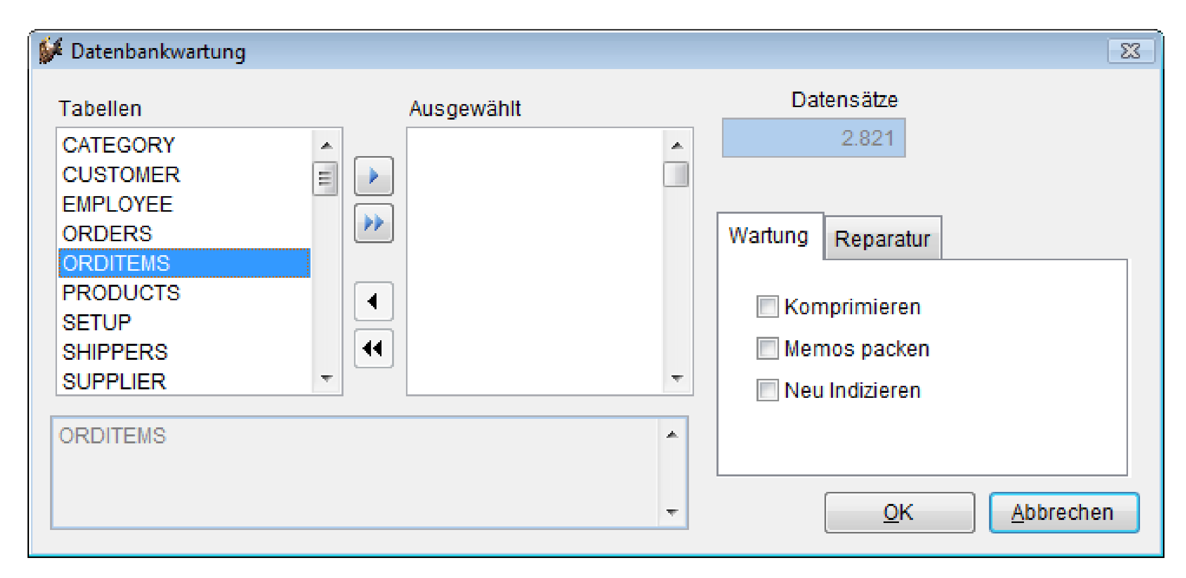
Nach der erfolgreichen Anmeldung eines Benutzers wird ein global sichtbares Objekt mit dem Namen *goUser* angelegt. Für alle Felder des aktuellen Benutzer-Datensatzes (aus der Tabelle *Vfxusr.dbf*) der dem angemeldeten Benutzer gehört, wird dem Objekt *goUser* eine Eigenschaft hinzugefügt. Der Name der Eigenschaft entspricht dem Namen des Feldes in der Tabelle *Vfxusr.dbf*. Es ist an jeder Stelle im Programm möglich, den Wert dieser Eigenschaft abzufragen um zu entscheiden, ob ein Benutzer eine bestimmte Aktion ausführen darf. So kann z. B. die Auswahl eines Menüpunkts, das Öffnen eines Formulars oder das Bearbeiten eines Feldes auf einem Formular verhindert werden.

2.2.6. Fehlerprotokoll

Sollte es einmal zu einem Laufzeitfehler kommen, wird der Fehler in einer Messagebox angezeigt. Außerdem wird der Fehler in einer Tabelle protokolliert. Dabei werden der Name des aktuellen Benutzers, Datum, Uhrzeit, der Status aller geöffneten Tabellen sowie die Ausgabe von List Memory gespeichert. Weitere Eigenschaften der Behandlung von Laufzeitfehlern können über Eigenschaften des Anwendungsobjekts eingestellt werden.

2.2.7. Datenbankwartung

Über den Menüpunkt System – Datenbankwartung wird ein Formular mit einem Mover-Dialog angezeigt.



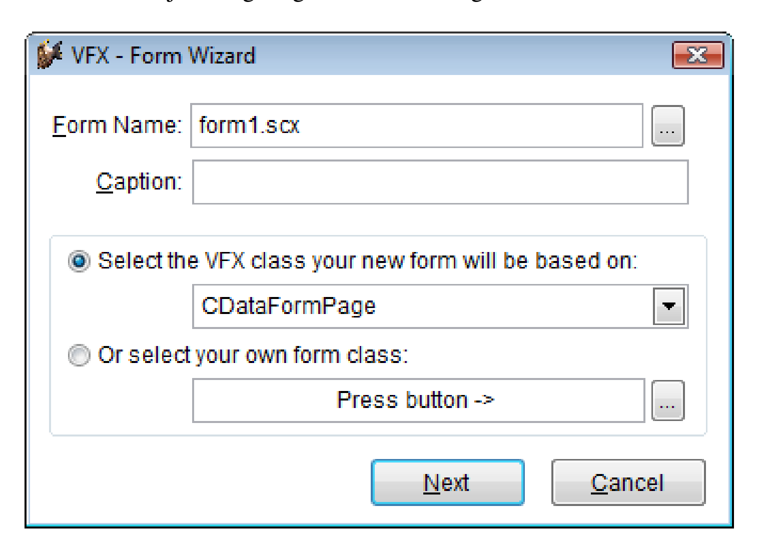
Hier können Tabellen gepackt oder indiziert werden.

2.2.8. Info-Dialog

Ein Standard-Info-Dialog ist in allen VFX-Anwendungen enthalten. Die angezeigten Parameter stammen aus einer Include-Datei, die beim Anlegen des Projektes erzeugt wurde.

2.3. Erstellen eines Formulars mit dem VFX – Form Wizard

Mit Hilfe des VFX – Form Wizard wird ein neues Formular auf der Basis einer VFX-Formularklasse angelegt und in das Projekt eingetragen. Die am häufigsten verwendete Formularklasse ist die Klasse *CDataFormPage*.



2.4. VFX – Data Environment Builder

Im nächsten Schritt wird in jedem VFX Form Builder die Datenumgebung bearbeitet. Die von dem Formular zu verwendenden Tabellen oder Ansichten sind in der Datenumgebung einzutragen.

Der Datenumgebung können Tabellen, Ansichten oder bestehende CursorAdapter-Klassen hinzugefügt werden oder auch neue CursorAdapter-Klassen erstellt werden. Mit einem Klick auf die Schaltfläche *Add* können bestehende Tabellen oder Ansichten der Datenumgebung hinzugefügt werden. Der VFP-Dialog zur Auswahl von Tabellen und Ansichten wird geöffnet. Wenn ein Cursor in der Datenumgebung auf einer Tabelle basiert, kann in der Spalte Order ein Index der Tabelle gewählt werden. Wenn ein Cursor in der Datenumgebung auf einer CursorAdapter-Klasse basiert und für diesen CursorAdapter Indexschlüssel definiert wurden, kann aus diesen Indexschlüsseln in der Spalte Order ebenfalls ausgewählt werden.

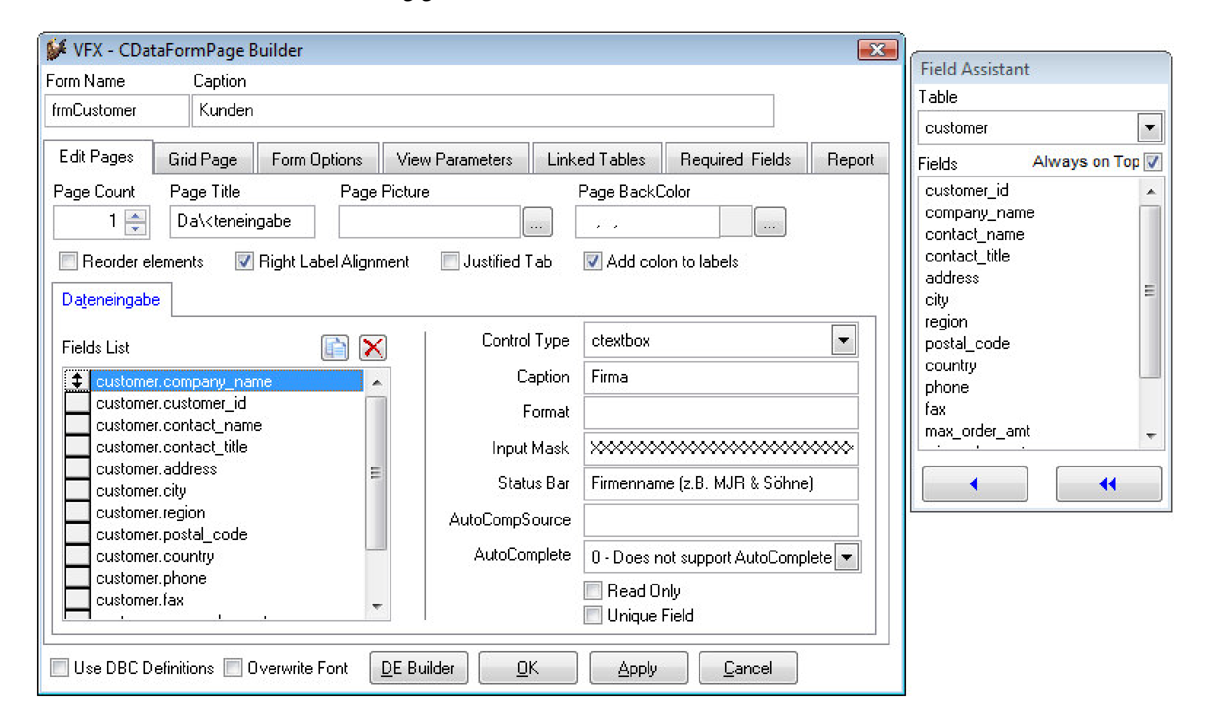
Für ein einfaches Formular zur Bearbeitung von Daten aus einer Tabelle ist es ausreichend diese Tabelle der Datenumgebung hinzuzufügen. Anschließend können dem Formular mit dem VFX – Form Builder Steuer-elemente hinzugefügt werden.

Der VFX – Form Builder liest die Datenumgebung aus und stellt die Felder der Tabellen zur Auswahl um Steu-erelemente zu erstellen. Zur Laufzeit wird die Datenumgebung ebenfalls ausgelesen um die Tabellen zu ermit-teln, für die ein *Tableupdate* bzw. *Tablerevert* durchgeführt werden muss.

2.5. Der VFX – Form Builder

Mit dem Form Builder werden die für das Formular benötigten Steuerelemente erstellt. Für jedes Steuerelement können dabei die zugrunde liegende VFX-Klasse gewählt sowie viele Eigenschaften eingestellt werden.

Beim ersten Erstellen des Formulars wird automatisch ein Eintrag in der Tabelle *Vfxfopen.dbf* angelegt, sodass das Formular über den Öffnen-Dialog gestartet werden kann.



Der VFX – Form Builder ist voll reentrant. Das heißt, man kann den Builder beliebig oft aufrufen um Einstellungen an einem Formular zu verändern. Es ist auch möglich das Formular von Hand mit VFP zu bearbeiten und anschließend wieder mit dem Form Builder zu arbeiten, ohne dass Einstellungen verloren gehen oder überschrieben werden.

2.6. Der VFX – CGrid Builder

Sollen nur Änderungen am Grid vorgenommen werden, braucht nicht der Form Builder verwendet zu werden. Mit dem VFX – Grid Builder können die Einstellungen des Grids verändert werden. Wie alle VFX Builder ist auch der Grid Builder reentrant.

2.7. Test

Das Formular kann direkt aus dem VFP Formular-Designer oder aus dem Projekt-Manager gestartet und getestet werden. Im *Init()*-Ereignis aller VFX-Formulare wird geprüft, ob das Anwendungsobjekt existiert. Falls dieses nicht vorhanden ist, wurde das Formular direkt aus dem Projekt-Manager gestartet und VFX stellt selbstständig die Umgebung her, um das Formular laufen zu lassen. Dabei wird auch die Hauptsymbolleiste instanziiert und kann für die Bedienung des Formulars verwendet werden.

Natürlich ist es auch möglich das Projekt über das Hauptprogramm *Vfxmain.prg* zu starten. Das Formular kann dann über den Öffnen-Dialog gestartet werden.

3. Einführung

3.1. Überblick

Zum Betrieb von Visual Extend 11.0 ist Visual FoxPro 9.0 erforderlich.

Visual Extend 11.0 stellt eine umfassende Entwicklungsumgebung für Softwareentwickler dar, die mit Microsoft Visual FoxPro 9.0 oder einer neueren Version arbeiten. Visual Extend beinhaltet Builder, die den Softwareentwickler bei seiner täglichen Arbeit unterstützen und so die Entwicklerproduktivität drastisch steigern. Dies, ohne jegliche Einbußen bezüglich Flexibilität oder Leistungsfähigkeit in Kauf nehmen zu müssen. Visual Extend macht aus Visual FoxPro ein echtes Rapid Application Development Tool, dies sowohl für Desktop- als auch für Client/Server-Datenbank- Anwendungsentwicklungen.

Visual FoxPro ist ein exzellentes Entwicklungswerkzeug. Dank der Objektorientierung und der OLE-Technologie wird der Traum eines jeden Softwareentwicklers nach einfachster Wiederverwendung von eigenen oder fremden Softwaremodulen Wirklichkeit. Das Erstellen einer eigenen Entwicklungsumgebung stellt jedoch ein größeres Unterfangen dar, welches sich heutzutage immer weniger Softwareentwickler wirklich leisten können. Es ist nicht nur schwierig, eine stabile Klassenbibliothek für alle Anwendungen zu entwickeln, es wäre auch sehr zeitaufwendig, die Klassen manuell einzusetzen und alle Eigenschaften und Methoden über das Eigenschaftsfenster während der Entwicklung einer neuen Anwendung zu bearbeiten.

Visual Extend für Visual FoxPro füllt exakt diese Lücke und stellt eine vollständige Anwendungsentwicklungsumgebung für Visual FoxPro Softwareentwickler dar. Dank des durchdachten, modularen Designs von Visual Extend kann der Softwareentwickler jederzeit selbst entscheiden, ob er die gesamte Entwicklungsphilosophie von Visual Extend verwenden oder nur ausgewählte Teile daraus für die Erstellung seiner eigenen Anwendungen übernehmen will. Die Objektorientierung von Visual Extend erlaubt dem Entwickler Unterklassen aller Visual Extend Klassen zu erstellen, um so die Entwicklungsumgebung noch besser seinen spezifischen Bedürfnissen anzupassen.

Visual Extend ist weit mehr als nur eine Sammlung von Klassenbibliotheken. Vielmehr beinhaltet Visual Extend neben leistungsfähigen Klassenbibliotheken ebenso leistungsfähige Builder, um einen maximalen Produktivitätsgewinn zu erzielen. Visual Extend besteht aus den folgenden Hauptkomponenten:

- Modulare den Microsoft Standards entsprechende Klassenbibliotheken zur umfassenden Unterstützung bei der Anwendungsentwicklung.
- Visual Extend Assistenten und voll wieder verwendbare Builder für Anwendung, Formular, Grid, Child-Grid, Auswahlliste, Auswahltextfeld, 1:n-Formulare und vieles andere mehr.
- Weitere Visual Extend Entwickler-Produktivitätswerkzeuge wie das Entwicklermenü, die VFX Task Pane, der VFX – Base Class Switcher und der Visual Object Name Picker.

3.2. Eigenschaften von mit Visual Extend erstellten Anwendungen

Anwendungen, die mit Visual FoxPro und der Software-Entwicklungsumgebung Visual Extend entwickelt wurden, haben die folgenden Eigenschaften:

- Bereit zur Office-Compatible-Zertifizierung.
- Standard-Symbolleiste und optionale individuelle Symbolleiste für jedes Formular.
- Unterstützung von XP-Themes in allen Steuerelementen.
- Hot Tracking von Schaltflächen in Symbolleisten.
- Icons in Menüs.
- Navigieren, Suchen, Neu, Kopieren, Bearbeiten, Löschen als Optionen im Formular oder in der Symbolleiste.
- Multiinstanzfähige Formulare.
- Zuletzt aufgerufene Formulare im Menü Datei sowie aktuell geöffnete Formulare im Menü Fenster.
- Inkrementelle Suche inklusive automatischer Sortierung in allen VFX-Grids.
- Wechsel der Sortierung durch Doppelklick auf die Spaltenüberschrift in jedem VFX-Grid.
- Anzeige der aktuellen Sortierung in der Spaltenüberschrift, wahlweise auch farbliche Anzeige.
- Automatisches Speichern und Wiederherstellen der Größe und der Position von allen Formularen.
- Automatisches Speichern und Wiederherstellen aller Layoutänderungen und der Sortierfolge im Grid.

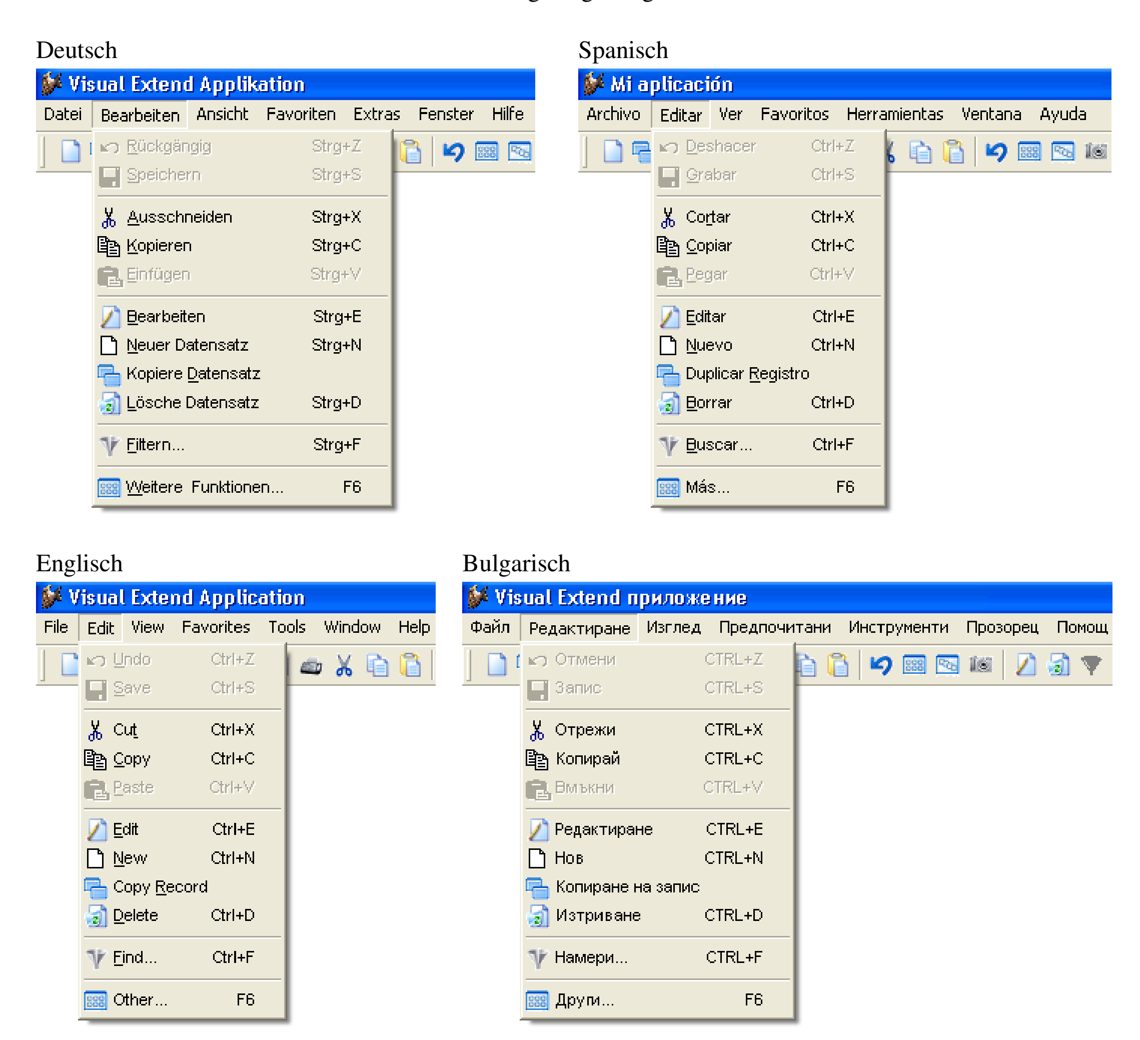
- Auswahllisten-Steuerelement mit automatischer Validierung.
- Auswahllisten-Formular mit inkrementeller Suche, automatischer Sortierung, Wechsel der Sortierung durch Doppelklick auf eine Spaltenüberschrift und Start des Bearbeitungsformulars mit der Möglichkeit neue Datensätze einzugeben.
- Automatisches Speichern und Wiederherstellen der Größe und Position von allen Auswahllisten-Formularen inklusive aller Layoutänderungen im Auswahllisten-Grid.
- Leistungsfähige Auswahllisten in Child-Grids.
- Benutzerverwaltung mit Kennwort-Verschlüsselung.
- Automatische Übernahme des Netzwerk-Anmeldenamens und Möglichkeit der automatischen Benutzeranmeldung.
- Verwaltung der Benutzerrechte mit Ansichts-, Bearbeitungs-, Neuanlage-, Kopier-, Druck- und Löschrecht auf Formularebene.
- Datenbankwartung für das Komprimieren und neu Indizieren von lokalen Tabellen sowie einer Option um defekte Datenbanken zu reparieren.
- Automatisches protokollieren aller Laufzeitfehler.
- Infodialog.
- Benutzerfreundliche Mover-Dialoge für die einfache Auswahl mehrerer Elemente.
- Automatische Übernahme der Windows-Systemfarben.
- Favoriten-Menü.
- Öffnen-Formular im XP-Stil.
- Optionale „Active-Desktop“ Einzelklick-Benutzeroberfläche.
- Automatisches Erstellen von gedruckten Berichten basierend auf der Datenanzeige in einem Grid.
- Berichtsauswahl und –bearbeitungsdialog.
- Unterstützung mehrerer Datenbanken mit der Möglichkeit die Datenbank zur Laufzeit zu wechseln.
- Automatische Aktualisierung der Strukturen der Kundendatenbank für VFP- und SQL Server-Datenbanken.
- Optionales Bearbeitungsprotokoll zur Verfolgung der Datenbearbeitung.
- Die Microsoft Agenten können zur Gestaltung der Benutzeroberfläche verwendet werden.
- Automatischer Ausdruck des Bildschirminhalts.
- Es können mehrsprachige Anwendungen erstellt werden.

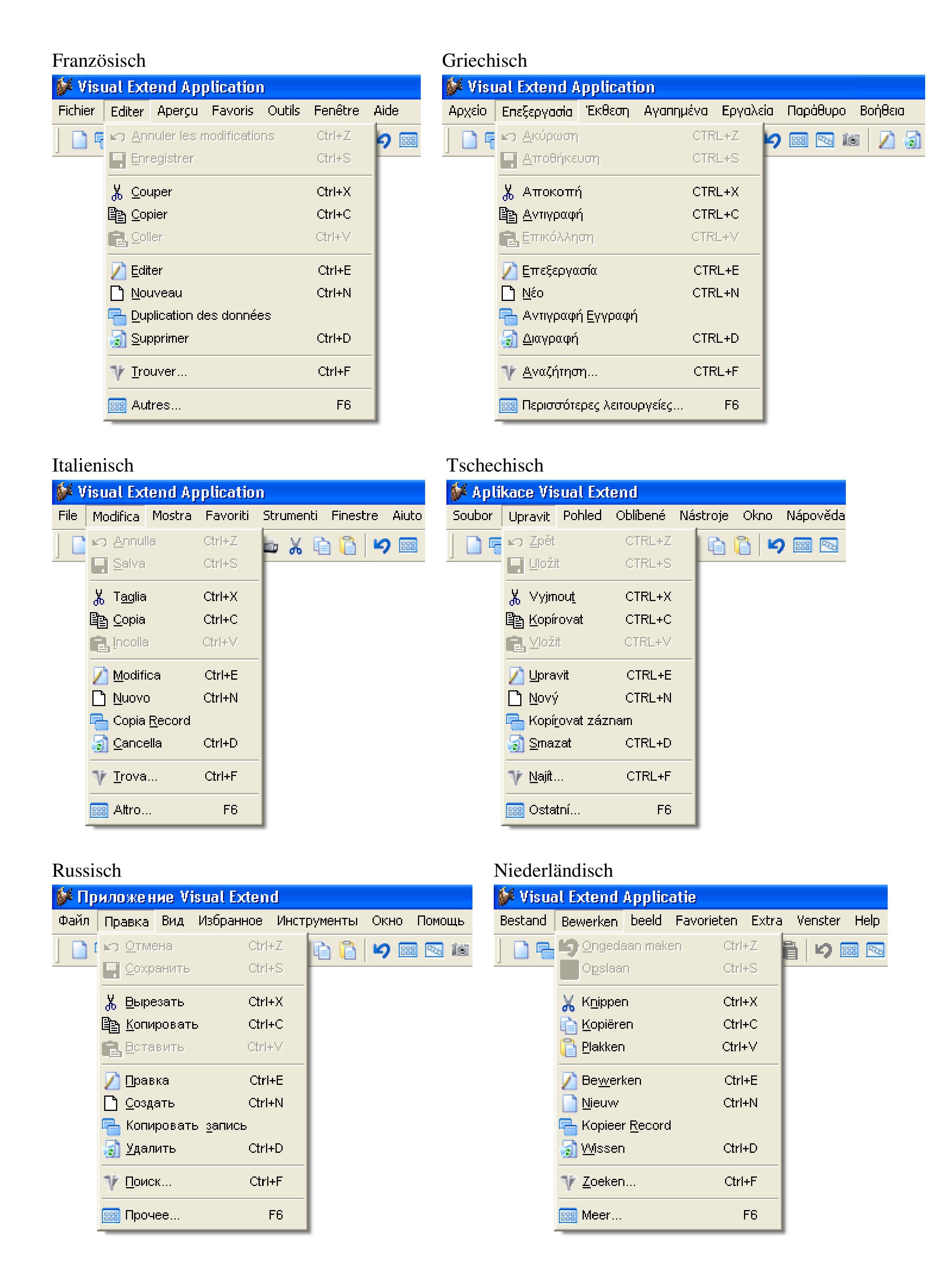
3.3. Leistungsmerkmale für Entwickler

Softwareentwickler werden die folgenden Visual Extend-Merkmale besonders zu schätzen wissen:

- Anwendungs-Assistent für das automatische Erstellen von neuen Anwendungen in der Sprache Ihrer Wahl. Nach nur wenigen Sekunden ist Ihre lauffähige Visual Extend-Anwendung vorbereitet!
- Volle Widerverwendbarkeit von allen VFX-Buildern (Formular-Builder, 1:n-Formular-Builder, Table Form-Builder, Grid-Builder, Child-Grid-Builder, Auswahltextbox-Builder), die es vereinfachen, Änderungen an mit den VFX-Buildern erstellten Formularen durchzuführen!
- Benutzen Sie die Visual FoxPro-Entwicklungsumgebung wann immer Sie wollen, ohne die Widerverwendbarkeit der VFX-Builder zu verlieren, solange Sie alle Steuerelemente mit Hilfe der VFX-Builder hinzufügen bzw. entfernen!
- Builder für Standardformulare inklusive Parent/Child-Technik (aufrufen und aufgerufen von).
- Builder für leistungsfähige Grids.
- Builder für jeden Bedarf an Auswahllisten.
- Builder für klassische sowie fortgeschrittene 1:n-Formulare mit mehrseitiger Bearbeitung der Haupttabelle sowie mehrseitiger Bearbeitung für mehrere Child-Tabellen in einem Formular.
- Alle Builder lesen die vorhandenen Feldbeschreibungen und andere Eigenschaften aus der Datenumgebung.
- Die Formular-Builder passen die Längen der Textfeld-Steuerelemente den Größen der zugrunde liegenden Felder an.
- Die VFX-Formular-Builder sind auf eigenen, von den VFX-Klassen abgeleiteten Klassen einsetzbar.
- Testen von Formularen direkt aus dem VFP Formular-Designer.
- Navigieren mit der Symbolleiste oder mit Navigations-Schaltflächen auf dem Formular oder mit Schaltflächenleisten innerhalb eines Formulars.
- Messagebox-Assistent.
- Task Pane Anwendungs-Manager.
- Einfache Änderungen an der Klasse des Anwendungsobjekts durch eine Ableitung in Appl.vcx.
- Einfaches Erstellen der anwendungsspezifischen Standard-Symbolleisten.

- Technik verbundener Parent/Child-Formulare.
- Die Entwicklungsumgebung stellt bereits alle Elemente der Benutzeroberfläche in den Sprachen bulgarisch, tschechisch, niederländisch, englisch, französisch, finnisch, deutsch, griechisch, italienisch, portugiesisch, russisch und spanisch zur Verfügung. Starten Sie eine neue Anwendung in der Sprache Ihrer Wahl, ohne ein Wort der Visual Extend Software-Entwicklungsumgebung übersetzen zu müssen.




Französisch
Visual Extend Application
Fichier Editer Aperçu Favoris Outils Fenêtre Aide
Annuler les modifications Ctrl+Z
Enregistrer Ctrl+S
Couper Ctrl+X
Copier Ctrl+C
Coller Ctrl+V
Editer Ctrl+E
Nouveau Ctrl+N
Duplication des données
Supprimer Ctrl+D
Trouver... Ctrl+F
Autres... F6
Griechisch
Visual Extend Application
Αρχείο Επεξεργασία Έκθεση Αγαπημένα Εργαλεία Παράθυρο Βοήθεια
Ακύρωση CTRL+Z
Αποθήκευση CTRL+S
Αποκοπή CTRL+X
Αντιγραφή CTRL+C
Επικόλληση CTRL+V
Επεξεργασία CTRL+E
Νέο CTRL+N
Αντιγραφή Εγγραφή
Διαγραφή CTRL+D
Αναζήτηση... CTRL+F
Περισσότερες λειτουργίες... F6
Italienisch
Visual Extend Application
File Modifica Mostra Favoriti Strumenti Finestre Aiuto
Annulla Ctrl+Z
Salva Ctrl+S
Taglia Ctrl+X
Copia Ctrl+C
Incolla Ctrl+V
Modifica Ctrl+E
Nuovo Ctrl+N
Copia Record
Cancella Ctrl+D
Trova... Ctrl+F
Altro... F6
Tschechisch
Aplikace Visual Extend
Soubor Upravit Pohled Oblíbené Nástroje Okno Nápověda
Zpět CTRL+Z
Uložit CTRL+S
Vyjmout CTRL+X
Kopírovat CTRL+C
Vložit CTRL+V
Upravit CTRL+E
Nový CTRL+N
Kopírovat záznam
Smazat CTRL+D
Najít... CTRL+F
Ostatní... F6
Russisch
Приложение Visual Extend
Файл Правка Вид Избранное Инструменты Окно Помощь
Отмена Ctrl+Z
Сохранить Ctrl+S
Вырезать Ctrl+X
Копировать Ctrl+C
Вставить Ctrl+V
Правка Ctrl+E
Создать Ctrl+N
Копировать запись
Удалить Ctrl+D
Поиск... Ctrl+F
Прочее... F6
Niederländisch
Visual Extend Applicatie
Bestand Bewerken beeld Favorieten Extra Venster Help
Ongedaan maken Ctrl+Z
Opslaan Ctrl+S
Knippen Ctrl+X
Kopiëren Ctrl+C
Plakken Ctrl+V
Bewerken Ctrl+E
Nieuw Ctrl+N
Kopieer Record
Wissen Ctrl+D
Zoeken... Ctrl+F
Meer... F6



VFX hilft Ihnen, Ihre Visual FoxPro-Anwendungen in einer höheren Qualität und wesentlich schneller als bisher zu entwickeln. Ihre Entwickler-Produktivität steigert sich dramatisch. Und das alles, ohne irgendwelche Einbußen bezüglich der Flexibilität von Visual FoxPro in Kauf nehmen zu müssen. Produktiver als je zuvor mit Visual Extend für Visual FoxPro**!**

4. Leistungsumfang

4.1. VFX-Klassenbibliotheken

Sie finden die Klassenbibliotheken im Ordner *\VFX110\LIB*. Um eine detaillierte Beschreibung aller Dateien der Klassenbibliotheken mit allen Klassen, Eigenschaften und Methoden zu bekommen, lesen Sie bitte in der VFX Technische Referenz nach. Die Technische Referenz ist eine Windows-Hilfedatei.

4.2. VFX-Assistenten und Builder

Alle VFX-Assistenten und Builder befinden sich im Ordner *\VFX110\BUILDER*:

Assistent	Datei	Beschreibung
VFX Menü	VFXMNU.APP	Richtet einen speziellen Menüpunkt in Ihrem Visual FoxPro Menü ein. Von diesem Menü aus können Sie den VFX-Anwendungs-Assistenten und weitere VFX-Assistenten aufrufen. **Tipp:** Wenn Sie die Installationsanleitung befolgt haben, wird dieses Menü automatisch geladen wenn Sie VFP starten.
VFX-Assistenten und Builder	VFXBLDR.APP	Die folgenden VFX-Assistenten und Builder helfen Ihnen bei der Erstellung von professionellen Visual FoxPro-Anwendungen in Rekordzeit: √ Anwendungs-Assistent für die Erstellung einer neuen Anwendung √ Formular-Assistent für die Erstellung eines neuen Formulars √ Formular-Builder (inklusive mehrseitigen Formularen, **wieder verwendbar**) √ Grid-Builder (**wieder verwendbar**) √ Auswahllisten-Builder (**wieder verwendbar**) √ 1:n-Builder (inklusive mehrseitigen Seitenrahmen für die Haupttabelle und mehreren Seiten für die Child-Tabellen, **wieder verwendbar**) √ Child-Grid-Builder (**wieder verwendbar**) √ Auswahllisten-Builder für Auswahllisten innerhalb von Child-Grids (**wieder verwendbar**) *Wenn Sie die Installationsanweisungen befolgen, können Sie mittels rechter Maustaste den VFX-Builder aufrufen, nachdem Sie das entsprechende Objekt ausgewählt haben.*
VFX LangSetup Builder	LANGBLDR.APP	Automatisieren Sie die Erstellung des Codes für die *LangSetup()*-Methode. Dies ist eine sehr große Hilfe, wenn Sie mehrsprachige Anwendungen erstellen. ***Aufrufen können*** *Sie den LangSetup-Assistenten aus dem VFX-Menü oder indem Sie LANGBLDR.APP starten*
VFX Messagebox Builder	MSGBLDR.APP	Automatisieren Sie das Generieren von Messagebox-Dialogen und den zugehörigen Konstanten in den Include-Dateien. ***Aufrufen können*** *Sie den Messagebox-Assistenten aus dem VFX-Menü oder indem Sie MSGBLDR.APP starten.*
VFX Message Editor	MSGEDIT.APP	Automatisieren Sie die Lokalisierung von Meldungen und anderen Texten sowie das Generieren der entsprechenden Include-Dateien. ***Aufrufen können*** *Sie den Message-Editor aus dem VFX-Menü oder indem Sie MSGEDIT.APP starten.*
VFX Menu Designer	VMD.APP	Erstellen Sie professionelle Menüs, die alle Eigenschaften unterstützen, die mit VFP möglich sind. Der visuelle VFX Menü-Designer unterstützt sehr viel mehr Eigenschaften, als der VFP Menü-Designer. ***Aufrufen können*** *Sie den VFX Menü-Designer, indem Sie im VFP Projekt-Manager ein Menü zur Bearbeitung öffnen.*
VFX – AFP Wizard	VFXAFPWIZARD.APP	Erstellen Sie Internet-Anwendungen mit Formularen, die in ihrem Aussehen und ihrer Funktion den Formularen Ihrer VFX-Anwendung entsprechen. ***Aufrufen können*** *Sie den VFX – AFP Wizard direkt aus dem VFX Menü.*
Project Documenting	PDM.EXE	Der Projekt-Dokumentierungsassistent erstellt zu Ihrem VFX-Projekt ein umfangreiche technische Dokumentation im HTML-Format. ***Aufrufen können*** *Sie den Project Documenting Assistenten direkt aus dem VFX Menü.*

Alle VFX Formular-, Grid- und Auswahllisten-Builder sind voll wieder verwendbar! Das bedeutet, dass Sie diese Builder im Entwicklungszyklus beliebig oft aufrufen können, ohne zuvor eingegebene Einstellungen zu verlieren. Ebenso werden Änderungen Ihres Formulars, die Sie nach der Generierung mit dem Visual FoxPro Formular-Designer gemacht haben, von den VFX Buildern beim nächsten Aufruf eingelesen.

Durch die offene Architektur der VFX-Assistenten steht fortgeschrittenen Benutzern der von den Assistenten verwendete Code in der Tabelle *\VFX110\LIB\BUILDER\VFXCODE.DBF* zur Verfügung. Dadurch können Sie die Assistenten einfach Ihren eigenen Code verwenden lassen. **Achtung:** Änderungen in dieser Tabelle erfordern fortgeschrittenes Wissen über VFX.

ANMERKUNG: Benutzen Sie die VFX-Builder so lange wie möglich, um Steuerelemente hinzuzufügen oder zu entfernen (definiert durch die ausgewählten Felder). Dadurch profitieren Sie am meisten von der hohen Produktivität, den die Builder bieten!

4.3. VFX-Produktivitätswerkzeuge

Um Ihre Arbeit mit VFX noch produktiver werden zu lassen, stehen Ihnen weitere nützliche Produktivitätswerkzeuge zur Verfügung:

Werkzeug	Datei	Beschreibung
VFX Task Pane	VFXTASKPANE.XML	Die VFX Task Pane erlaubt Ihnen ein problemloses Wechseln zwischen verschiedenen Projekten. Die Tabelle, die die aktuellen Referenzen zu Ihren Projekten speichert, ist *Vfxapp.dbf/cdx/fpt*. Diese Tabelle befindet sich im Ordner *C:\Dokumente und Einstellungen\All Users\Anwendungsdaten\dFPUG\Visual Extend\11.0*.
VFX Class Switcher	<VFXBLDR, aus dem VFX-Menü aufzurufen>	Ändert die Klasse aller Formulare. Ermöglicht ein einfaches Wechseln von Formularen mit Navigationsschaltflächen (z. B.: *CDataFormPageBar*) zu solchen ohne Navigationsschaltflächen (z. B.: *CDataFormPage*). Sie können mit dem Class Switcher auch die Klasse eines selektierten Steuerelementes ändern.
VFX Object Name Picker	<VFXBLDR, aus dem VFX-Menü aufzurufen>	Kopiert die vollständige Referenz des aktuell ausgewählten Steuerelements in die Zwischenablage. Das ist manchmal sehr nützlich, da visueller als die VFP-Objektliste, die Sie mit der rechten Maustaste in einem Codefenster öffnen können.

4.4. Weitere Entwicklerwerkzeuge

Zusätzlich zu den schon in früheren VFX-Versionen vorhandenen Buildern stehen in VFX 11.0 neue Power Builder für folgende Klassen zur Verfügung:

- *CTreeViewForm*
- *CTreeViewOneToMany*
- *CPickAlternate*
- *CPickAlterTextbox*

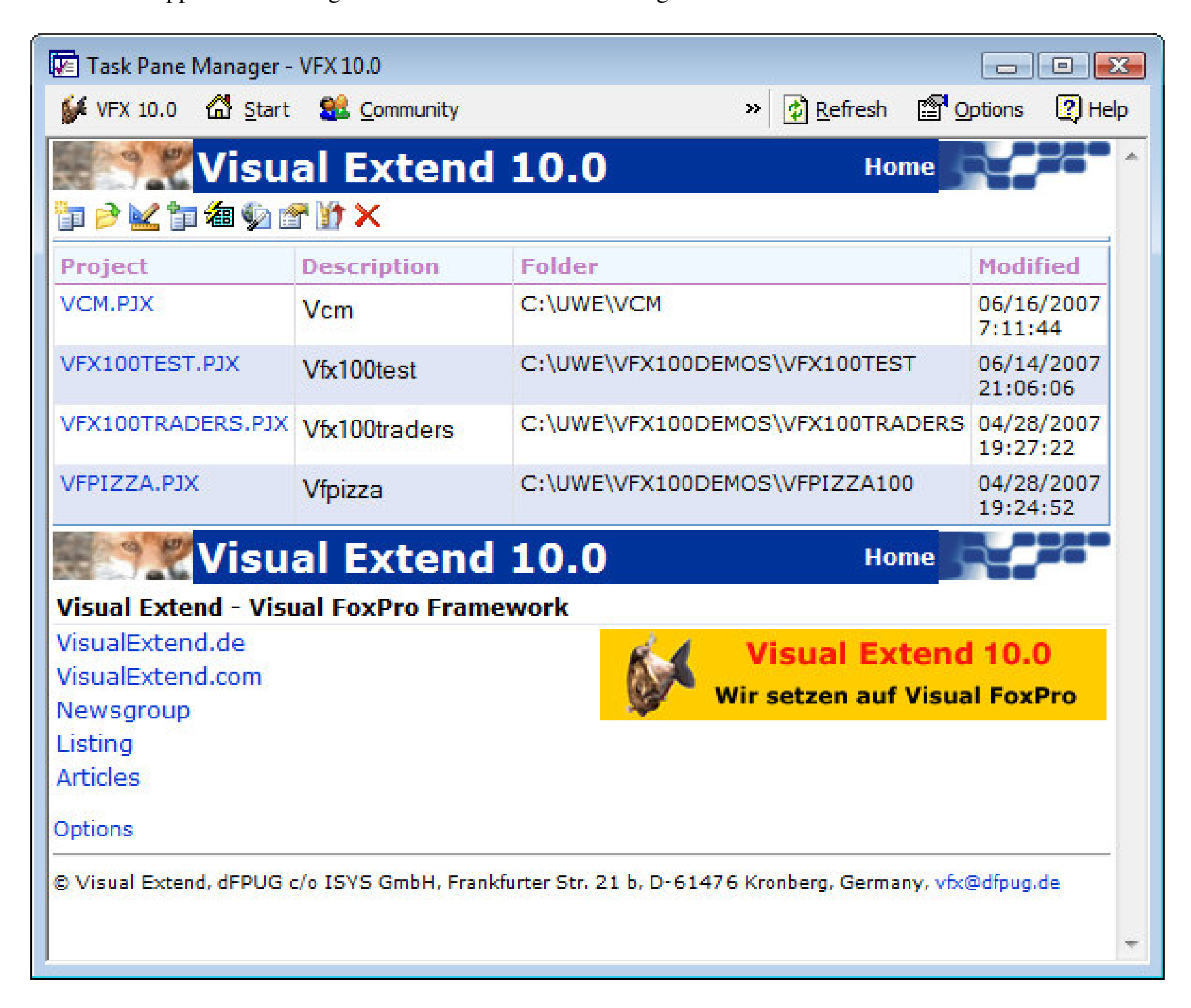
Zur weiteren Unterstützung gibt es die neuen bzw. überarbeiteten Assistenten:

- Define Activation Rules – Einstellen der Systemeigenschaften, die zur Produktaktivierung verwendet werden sollen sowie der möglichen Benutzerrechte.
- Create Activation Key – Erstellen eines Aktivierungsschlüssels anhand des Installationsschlüssels des Kunden.
- Customer List – Verwaltung von Kundendaten und Aktivierungsschlüsseln.
- Manage Application Updates – Verwaltung von Aktualisierungen der Anwendung über das Internet.
- Metadata Wizard – Zum Anlegen und aktualisieren von SQL Server-Datenbanken beim Kunden.
- Manage Config.vfx – Bearbeitung des Datenzugriffs.
- Cursor Adapter Wizard – Automatische Erstellung von CursorAdaptern zu allen Tabellen einer Datenbank.
- Audit Trigger Wizard – Erstellen von Triggern für ausgewählte Tabellen.
- Manage Vfxsys.dbf – Verwaltung der Tabelle Vfxsys.dbf mit teilweise verschlüsseltem Inhalt.
- VFX –AFP Wizard – Generierung von AFP-Seiten aus VFX-Formularen.

- Update Project Wizard – Aktualisierung von vorhandenen VFX-Projekten auf den aktuellen Build oder die aktuelle Version.
- Project Documenting – Erstellen einer technischen Dokumentation imHTML-Format.
- Project Toolbox – Hinzufügen der Klassen des aktuellen Projekts zur VFP Toolbox.
- Parent/Child Builder – Verwaltung der Beziehungen zwischen Parent- und Child-Formularen.
- VMD (Visual Extend Menu Designer)

4.5. VFX 11.0 Task Pane

Der VFX – Application Manager ist in die VFP Task Pane integriert.



Über die Symbolleiste stehen folgende Funktionen zur Verfügung:

New Project Startet den VFX – Application Wizard.

Open Project Öffnet ein VFP-Projekt und stellt den aktuellen Pfad auf den Projektordner.

Modify Project Öffnet das in der VFX 11.0 Task Pane selektierte Projekt und stellt den aktuellen Pfad auf den Projektordner.

Add Project Fügt ein vorhandenes VFP-Projekt der VFX 11.0 Task Pane hinzu.

Rebuild Neu kompilieren aller Dateien des in der VFX 11.0 Task Pane selektierten Projekts. Das Projekt wird nach dem kompilieren zur Bearbeitung geöffnet.

Properties Start der VFX – Project Properties zum in der VFX 11.0 Task Pane selektierten Projekt.

Project Backup Erstellt eine Zip-Datei vom selektierten Projekt.

Delete Entfernt das selektierte Projekt aus der VFX 11.0 Task Pane.

Mit einem einfachen Mausklick kann von einem Projekt eine Sicherungskopie in eine Zip-Datei erstellt werden.

Mit einem Klick auf das Symbol wird die Sicherung gestartet. Wenn das Projekt zu diesem Zeitpunkt geöffnet ist, wird es vor Beginn der Sicherung geschlossen.

5. Installation

5.1. Hardware- und Software-Anforderungen

Da es sich bei Visual Extend um eine Erweiterung zu Microsoft Visual FoxPro 9.0 handelt, benötigen Sie eine Hard- und Softwareumgebung, auf der Visual FoxPro 9.0 eingesetzt werden kann. Lesen Sie bitte bei den Systemanforderungen zu Microsoft Visual FoxPro nach.

5.2. Die Installation von VFX

Starten Sie das Installationsprogramm mit dem Namen *VFX110Setup.exe* und folgen Sie den Anweisungen auf dem Bildschirm.

Installieren Sie VFX 11.0 in einen neuen Ordner. Installieren Sie VFX 11.0 nicht in den Ordner, in dem sich eine frühere Version von VFX befindet!

Nach der Installation von VFX haben Sie diese Ordnerstruktu im VFX-Ordner:

Name	Änderungsdatum	Typ	Größe
bitmap	14.06.2007 21:04	Dateiordner	
Builder	20.06.2007 13:56	Dateiordner	
data	14.06.2007 21:04	Dateiordner	
form	14.06.2007 21:04	Dateiordner	
help	14.06.2007 21:04	Dateiordner	
include	14.06.2007 21:04	Dateiordner	
lib	26.06.2007 18:38	Dateiordner	
loader	14.06.2007 21:04	Dateiordner	
menu	14.06.2007 21:04	Dateiordner	
program	14.06.2007 21:04	Dateiordner	
regdata	18.06.2007 19:16	Dateiordner	
registerdll	14.06.2007 21:04	Dateiordner	
report	14.06.2007 21:04	Dateiordner	
reportprocessing	14.06.2007 21:04	Dateiordner	
config.fpw	23.03.2005 13:01	FPW-Datei	1 KB
pjhook.VCT	21.04.2007 20:13	Microsoft Visual FoxPro Class Library	21 KB
pjhook.vcx	21.04.2007 20:13	Microsoft Visual FoxPro Class Library	2 KB
vfx.fll	21.04.2007 18:01	FLL-Datei	104 KB
vfx.PJT	14.06.2007 21:04	Microsoft Visual FoxPro Project	277 KB
vfx.PJX	14.06.2007 21:04	Microsoft Visual FoxPro Project	57 KB
vfxapprights.dbf	17.12.2005 00:41	Microsoft Visual FoxPro Table	2 KB
vfxclass.DBF	25.01.2007 11:54	Microsoft Visual FoxPro Table	98 KB
vfxclass.FPT	06.07.2006 13:15	Microsoft Visual FoxPro Table	30 KB
vfxhelp.cdx	23.03.2005 13:02	Microsoft Visual FoxPro Index	6 KB
vfxhelp.dbf	22.05.2006 12:36	Microsoft Visual FoxPro Table	1 KB
vfxhelp.fpt	23.03.2005 13:02	Microsoft Visual FoxPro Table	1 KB
vfxpath.cdx	17.12.2005 00:41	Microsoft Visual FoxPro Index	5 KB
vfxpath.dbf	17.12.2005 00:41	Microsoft Visual FoxPro Table	1 KB

Der VFX-Ordner dient als zentraler Speicherplatz aller VFX-Komponenten und ist die Basis aller Projekte, die Sie mit dem VFX-Anwendungs-Assistenten erstellen, wie später in diesem Dokument beschrieben ist.

HINWEIS: Arbeiten Sie in diesem Projekt nicht direkt. Es ist NICHT für die direkte Bearbeitung gedacht. Verwenden Sie den Anwendungs-Assistenten, um ein neues Projekt zu erstellen.

5.3. Registrierung und Aktivierung von VFX 11.0

Wie bisherige Versionen von VFX ist auch VFX 11.0 über eine Produktaktivierung geschützt. Die Aktivierung von VFX 11.0 erfolgt mit einem Web Service. Der Vorteil ist, dass der Aktivierungsschlüssel unmittelbar an den Entwickler-PC gesendet wird und manuelle Tätigkeiten bei zur Eingabe des Schlüssels entfallen.

VFX 11.0 hat einen Software-Kopierschutz. Nach der Installation, beim ersten Start eines VFX-Builders oder des VFX-Menüs wird ein Registrierungsassistent angezeigt. Bitte füllen Sie alle erforderlichen Eingabefelder aus und klicken Sie auf die Schaltfläche „Register Online". Ihre persönlichen Daten werden über das Internet an den VFX-Registrierungs-Internet-Servers übertragen. Als Antwort erhalten Sie von dem Server einen Aktivierungsschlüssel, der auf der Festplatte Ihres Computers gespeichert wird. Der Aktivierungsschlüssel ist für 30 Tage gültig. In dieser Zeit können Sie den vollen Funktionsumfang von VFX testen.

Sollte Ihnen die Aktivierung über das Internet nicht direkt möglich sein, können Sie auf der Website

http://www.visualextend.de

einen Aktivierungsschlüssel bestellen. Sie bekommen den Aktivierungsschlüssel dann per E-Mail zugesendet.

Wenn VFX 11.0 mit einem 30-Tage-Testschlüssel betrieben wird, wird in einem Dialog die Restlaufzeit in Tagen angezeigt. Über die Schaltfläche *Buy VFX* wird die Website von Visual Extend angezeigt und es kann online eine Lizenz erworben werden. Nach Zahlungseingang erhalten Sie einen unbefristet gültigen Aktivierungsschlüssel per E-Mail zugestellt.

Beachten Sie, dass Sie die Installation von VFX nicht von einem PC auf einen anderen PC kopieren können ohne einen neuen Aktivierungsschlüssel anfordern zu müssen. Ihre Registrierungsnummer wird aus den Daten Ihres PCs ermittelt und ist einmalig. Jeder VFX-Benutzer hat eine andere, einmalige Registrierungsnummer und muss sich daher online registrieren, um den Aktivierungsschlüssel zu bekommen. Erst dann ist die Arbeit mit den VFX-Buildern möglich.

Wir hoffen, dass Sie den Software-basierten Schutz begrüßen und heißen Sie willkommen zur nächsten Generation von VFX. Dem besten VFX, das es je gab!

5.4. Einstellen der Visual FoxPro Umgebung für VFX

Sie müssen Microsoft Visual FoxPro 9.0 funktionsfähig installiert haben, bevor Sie die Arbeit mit VFX 11.0 beginnen können.

Als nächstes sollten Sie sicherstellen, dass das VFX 11.0-Menü jedes Mal automatisch erscheint, wenn Sie Ihr Visual FoxPro 9.0 starten. Starten Sie die Anwendung *Vfxmnu.app* direkt aus dem Windows-Explorer oder aus dem VFP-Befehlsfenster. Bei der Installation von VFX 11.0 werden Verknüpfungen zum Start von *Vfxmnu.app* im Windows Startmenü und auf dem Desktop angelegt.

Die Anwendung *Vfxmnu.app* befindet sich im Ordner *Builder* Ihrer VFX-Installation.

Wahlweise können Sie *Vfxmnu.app* auch über eine Startdatei ausführen.

Fügen Sie diese Zeile der Datei *CONFIG.FPW* in Ihrem VFP 9.0-Ordner hinzu:

ANMERKUNG: Wenn Sie keine Datei mit dem Namen *CONFIG.FPW* haben, können Sie diese Datei mit dem Editor anlegen.

```
command = do (HOME() + "vfx.prg")
```

Diese Zeile teilt VFP mit, dass das Programm VFX.PRG ausgeführt werden soll, wenn VFP gestartet wird. In der Datei VFX.PRG (erstellen Sie diese Datei ebenfalls mit dem Editor und speichern Sie diese auch im VFP-Ordner) fügen Sie folgende Zeile hinzu:

```
do c:\programme\vfx110\builder\vfxmnu.app
```

Wir gehen hier davon aus, dass VFX im Ordner C:\Programme\VFX110 installiert ist. Passen Sie den Pfad ggf. an.

Beim Start des VFX-Menüs werden automatisch die folgenden Einstellungen in VFP gemacht:

- **Builder:** zeigt Sie auf den VFX-Anwendungs-Assistenten mit dem Namen *VFXBLDR.APP* im Ordner *\VFX110\BUILDER*.

- **Suchpfad:** *\VFX110\BUILDER* wird dem Suchpfad hinzugefügt.

Beim ersten Start von VFP nach der Installation von VFX 11.0 wird die VFX 11.0 Task Pane automatisch in die VFP Task Pane integriert.

Wichtiger Hinweis: Stellen Sie sicher, dass Sie sich immer im Ordner Ihrer Anwendung befinden! Benutzen Sie den Befehl *cd ?* im Befehlsfenster, um den aktuellen Pfad zu prüfen oder noch besser, verwenden Sie die VFX Task Pane für ein einfaches Wechseln zwischen den verschiedenen Projekten, ohne dass Sie den Ordner manuell ändern müssen. Wenn Sie sich in einem falschen Ordner befinden, wird Visual FoxPro unter Umständen andere Include-Dateien oder Klassenbibliotheken verwenden als Sie erwarten.

Das beste Werkzeug um zwischen Projekten zu wechseln, ist die VFX 11.0 Task Pane. Sie können die Task Pane über den VFP-Menüpunkt *Extras, Task Pane* öffnen. Wir empfehlen die VFP Task Pane beim Start von VFP automatisch öffnen zu lassen. Wählen Sie hierzu im Task Pane Manager die Option „Open the Task Pane Manager when Visual FoxPro starts".

6. Erstellen einer Anwendung mit dem VFX – Application Wizard

6.1. Ziel

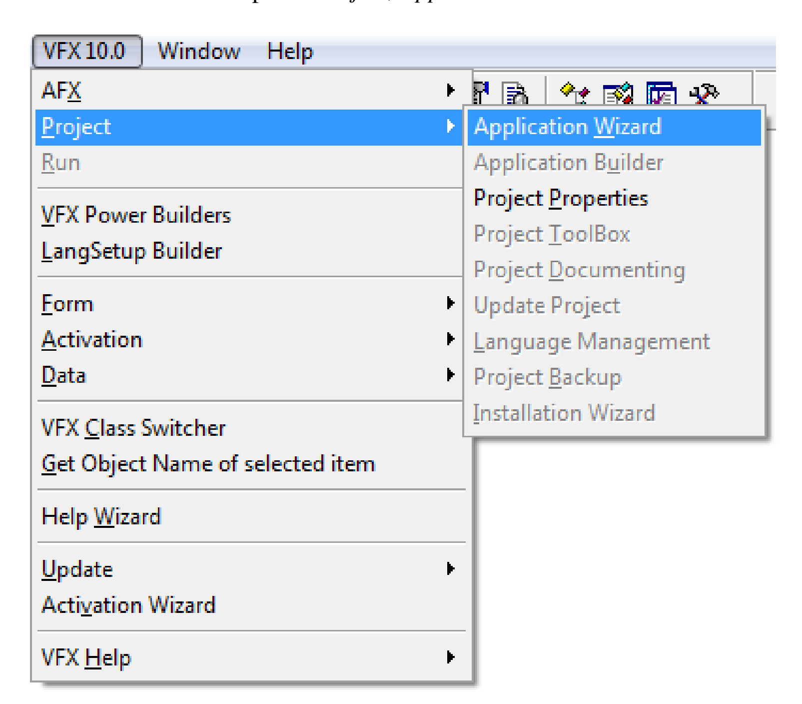
Wenn Sie ein neues Projekt beginnen, könnten Sie die ganze Ordnerstruktur von Hand erstellen, alle benötigten Dateien kopieren, wie etwa die Klassenbibliotheken, die Standardformulare, die Konfigurationsdateien, die Bilddateien usw. Hier greift der VFX-Anwendungs-Assistent ein: Er erstellt das gesamte Projekt in der Sprache Ihrer Wahl. Er stellt außerdem die wichtigsten Eigenschaften der Anwendungsklasse ein und erstellt Include-Dateien mit den wichtigsten Konstanten, um die manuelle Arbeit so weit wie möglich zu reduzieren.

6.2. Vorbereitung

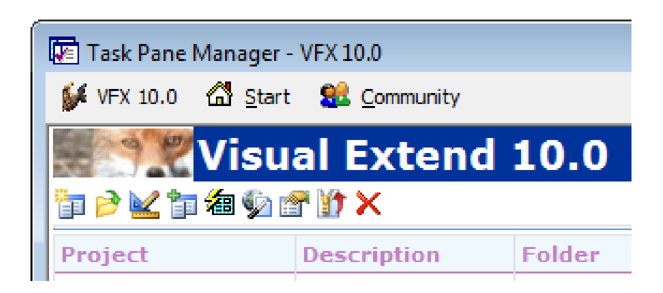
Schließen Sie alle Formulare und stellen Sie sicher, dass keine Klassenbibliotheken eines VFX-Projekts geöffnet sind. Am Besten beenden Sie Visual FoxPro und starten Sie erneut, bevor Sie den VFX-Anwendungs-Assistenten benutzen.

6.3. Der VFX – Application Wizard

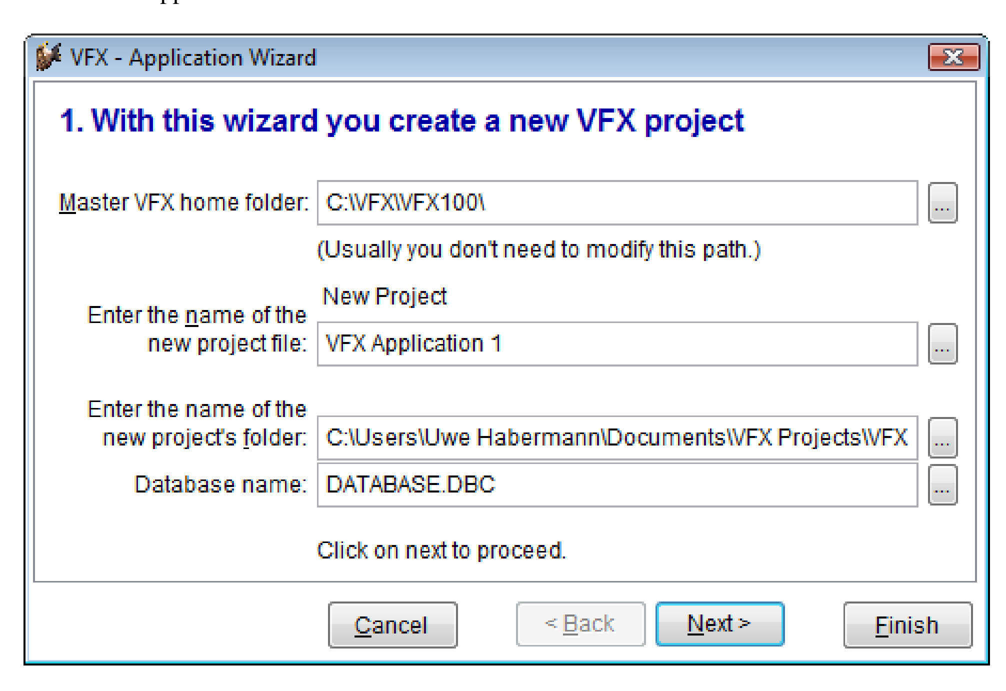
Wählen Sie den Menüpunkt *Project, Application Wizard* im VFX 11.0-Menü.



Oder starten Sie den *Application Wizard* aus der VFX Task Pane durch einen Klick auf das linke Symbol.



Der VFX – Application Wizard erscheint:



Die Einstellungen, die im VFX – Application Wizard gemacht werden, werden zur Verwendung in späteren Projekten gespeichert.

Geben Sie die folgenden Daten ein, bevor Sie eine neue Anwendung generieren lassen:

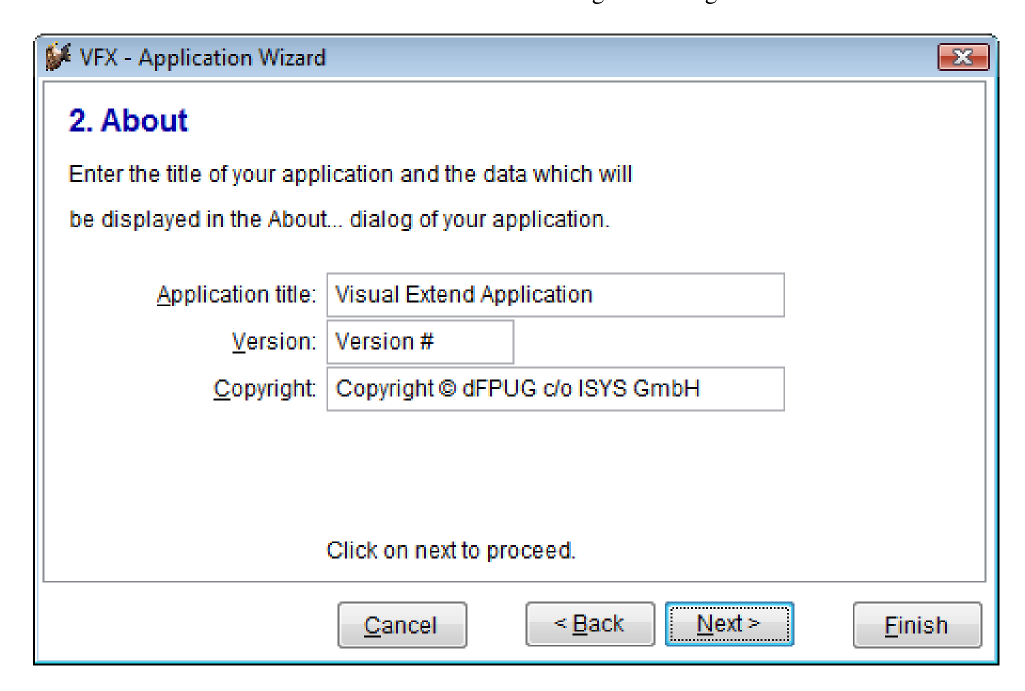
Master VFX home folder: Tragen Sie hier den VFX-Ordner ein, in dem sich Ihre VFX-Installation befindet. Normalerweise ist der vorgegebene Wert des Assistenten richtig und Sie brauchen keine Änderung zu machen.

Enter the name of the new project file: Geben Sie hier den Namen für Ihre neue Projektdatei ein. Fügen Sie keinen Pfad und keine Namenserweiterung hinzu. Geben Sie nur den Namen des neuen Projekts ein.

Enter the name of the new project's folder. Geben Sie den Ordner für Ihr neues Projekt ein. Wenn der Ordner noch nicht existiert, so wird er von dem VFX – Application Wizard erstellt. Der Standardpfad, in dem neue Projekte angelegt werden, ist „Eigene Dateien\VFX Projects\". Wenn ein anderer Pfad zum Erstellen eines Projektes gewählt wird, werden auch alle folgenden Projekte unter diesem Pfad gespeichert. Standardmäßig wird ein Projektordner mit dem Namen *VFX APPLICATION* gefolgt von einer fortlaufenden Nummer erstellt.

Database Name. Geben Sie den Namen Ihres Datenbank-Containers an (DBC). Geben Sie nur den Namen des Datenbank-Containers ohne Pfad und ohne Namenserweiterung ein. Wenn Ihre Anwendung auf eine Remote Datenquelle zugreifen soll und ausschließlich CursorAdapter für den Datenzugriff verwenden soll, können Sie dieses Feld leer lassen.

Auf der Seite mit dem Titel *2. About* machen Sie die folgenden Eingaben:

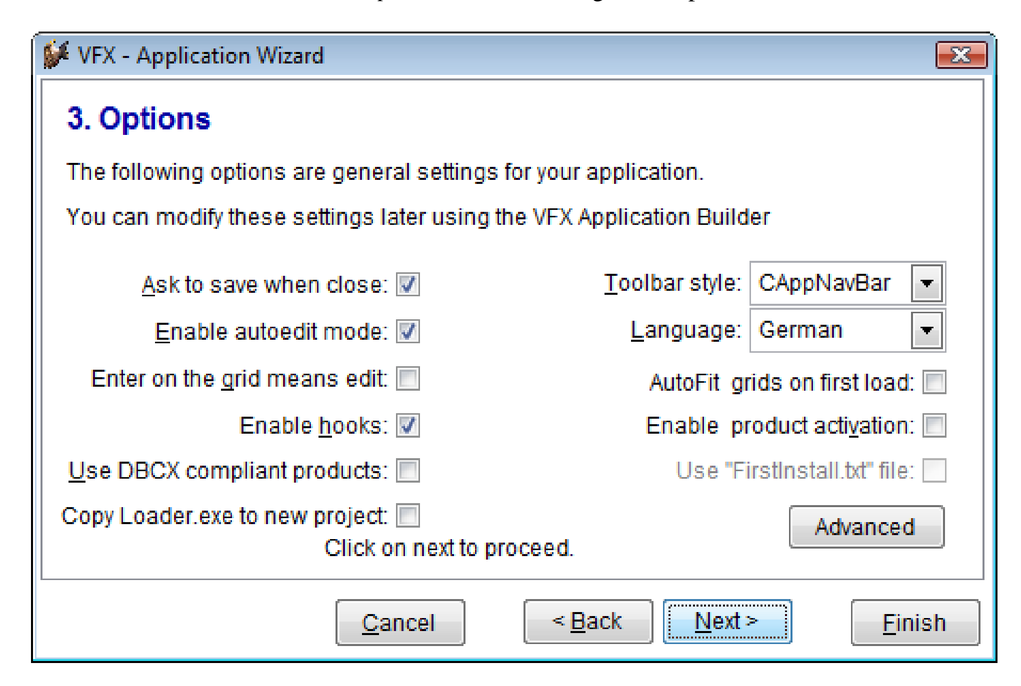


Application title: Geben Sie die Überschrift für das Hauptfenster Ihrer Anwendung an. Diese Überschrift wird als Konstante CAP_APPLICATION_TITLE in der Include-Datei USERTXT.H gespeichert.

Version: Geben Sie die Versionsnummer für den Infodialog Ihrer Anwendung ein. Die Nummer wird in der Konstante CAP_LBLVERSION in der Include-Datei USERTXT.H gespeichert.

Copyright: Geben Sie Ihre Copyright-Information für den Infodialog Ihrer Anwendung ein. Diese Information wird in der Konstante CAP_LBLCOPYRIGHTINFORMATION in der Include-Datei USERTXT.H gespeichert.

Auf der Seite mit dem Namen *3. Options* können Sie folgenden Optionen einstellen:



Ask to save when close: Die Auswahl dieser Option setzt den Wert der Eigenschaft *nAsktoSave* des Anwendungsobjekts auf 1. Diese Eigenschaft bestimmt das Verhalten von VFX wenn der Benutzer ein Formular schließt, nachdem er Änderungen am aktuellen Datensatz gemacht hat.

Enable autoedit mode. Die Auswahl dieser Option setzt den Wert der Eigenschaft *nAutoEditmode* des Anwendungsobjekts auf 1. Das bedeutet, dass der Benutzer jederzeit mit der Bearbeitung der Daten beginnen kann, ohne vorher in den Bearbeitungsmodus wechseln zu müssen.

Enter on the grid means edit: Die Auswahl dieser Option setzt den Wert der Eigenschaft *nEnterisEditinGrid* des Anwendungsobjekts auf 1. Das bedeutet, dass durch Drücken der Enter-Taste auf dem Grid einer Suchseite in den Bearbeitungsmodus gewechselt wird.

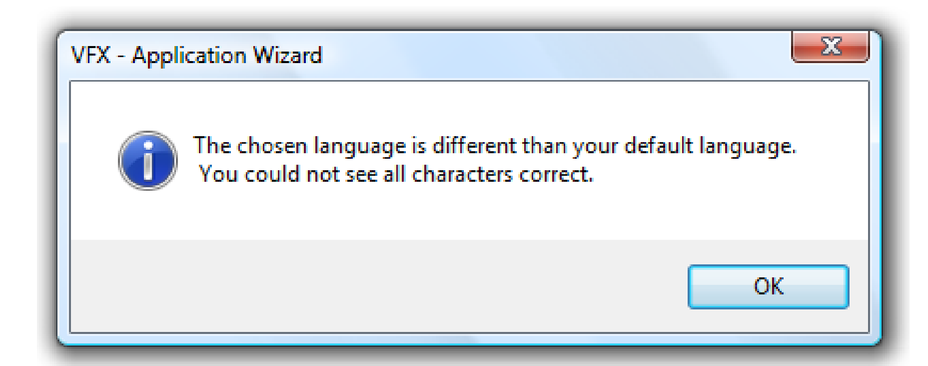
Enable hooks: Die Auswahl dieser Option setzt den Wert der Eigenschaft *nEnableHook* des Anwendungsobjekts auf 1. Das bedeutet, dass die Hooks aktiviert werden.

Use DBCX compliant products: Wenn der Stonefield Database Toolkit mit der zu erstellenden VFX-Anwendung eingesetzt werden soll, muss diese Option markiert werden.

Copy Loader.exe to new project: Zur Aktualisierung der Anwendung beim Kunden über das Internet wird die Datei Loader.exe benötigt. Wenn Sie das Loader-Projekt für Ihre Anwendung individuell anpassen möchten, markieren Sie diese Option.

Toolbar style: Wählen Sie hier die Symbolleistenklasse, die Sie verwenden wollen. *CAppNavBar* enthält Schaltflächen zur Bewegung des Datensatzzeigers und andere Schaltflächen zur Bearbeitung in der Standard-Symbolleiste. *CAppToolbar* enthält keine Schaltflächen zur Bewegung des Datensatzzeigers und zur Bearbeitung.

Language: Wählen Sie die gewünschte Sprache für Ihr neues Projekt. Bei der Auswahl einer Sprache für die generierte Anwendung prüft VFX die aktuellen Unicode-Einstellungen des Betriebssystems. Wenn die Zeichen der gewählten Sprache mit den aktuellen Einstellungen nicht angezeigt werden können, erscheint eine Warnung.



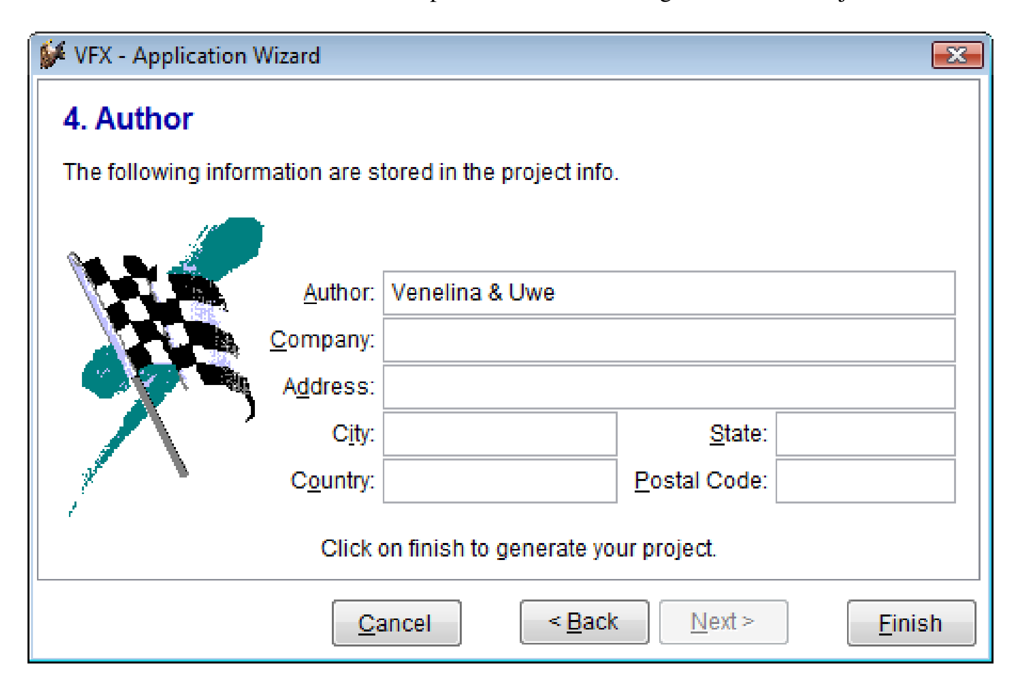
AutoFit grids on first load: Die Auswahl dieser Option setzt den Wert der Eigenschaft *nUseAutofit* des Anwendungsobjekts auf 1. Das bedeutet, dass bei Initialisierung von Grids das AutoFit-Ereignis aufgerufen wird.

Enable product activation: Die Auswahl dieser Option setzt den Wert der Eigenschaft *lUseActivation* des Anwendungsobjekts auf .T. Das bedeutet, dass die Anwendung eine Produktaktivierung erfordert.

Use „Firstinstall.txt" file: Die Auswahl dieser Option setzt den Wert der Eigenschaft *lActivationType* des Anwendungsobjekts auf .T. Das bedeutet, dass die Produktaktivierung die Datei *Firstinstall.txt* erfordert. Der Schutz Ihrer Anwendung wird dadurch weiter verbessert.

Advanced: Über diese Schaltfläche wird der VFX – Application Builder gestartet, der eine Vielzahl weiterer Einstellmöglichkeiten des Anwendungsobjekts bietet. Im unteren Teil dieses Dialogs wird ein Hilfetext mit einer Erklärung zur aktuellen Eigenschaft angezeigt.

Auf der Seite *4. Author* können Sie Ihre persönlichen Daten eingeben, um Ihr Projekt zu dokumentieren.



Diese Informationen werden in der Projektdatei gespeichert.

6.4. Erstellen des Projekts

Wenn Sie *Finish* auswählen, wird der VFX – Application Wizard ein neues Projekt entsprechend den von Ihnen eingegebenen Parametern erstellen. Dabei wird die Musteranwendung aus der VFX-Installation in den neuen Projektordner kopiert. Die Include-Dateien werden entsprechend der ausgewählten Sprache generiert. Anschließend wird das gesamte Projekt kompiliert, damit die in den Include-Dateien enthaltenen Konstanten zur Anwendung kommen. Eine abschließende Meldung zeigt an, dass Ihre neue Anwendung erfolgreich vorbereitet wurde.

ANMERKUNG: Da Sie sicher sofort mit der Arbeit an Ihrem neuen Projekt beginnen wollen, hat der VFX-Anwendungs-Assistent bereits automatisch den Standardordner auf den Startordner des neuen Projektes gesetzt. Um die Anwendung aus dem Projekt-Manager zu starten, wählen Sie das Hauptprogramm *VFXMAIN.PRG* und wählen Sie „ausführen“.

---

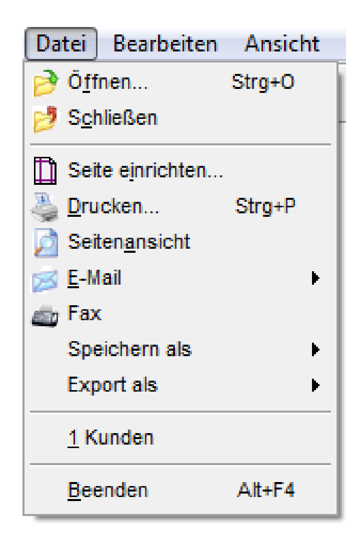
7. Diskussion der generierten VFX-Anwendung

Nach einer erfolgreichen Anwendungsgenerierung mit dem VFX-Anwendungs-Assistenten, haben Sie eine lauffähige Anwendung mit allem was eine neue Anwendung benötigt vom Menü, über die Standard-Symbolleiste, die Benutzerverwaltung, die Systemeinstellungen, Datenbankwartung, ein Laufzeitfehlerprotokoll bis hin zum Infodialog.

7.1. Office-kompatible Benutzeroberfläche

VFX erstellt Anwendungen, die nach dem *Office-Compatible*-Standard zertifiziert werden können.

7.1.1. Menü: Datei

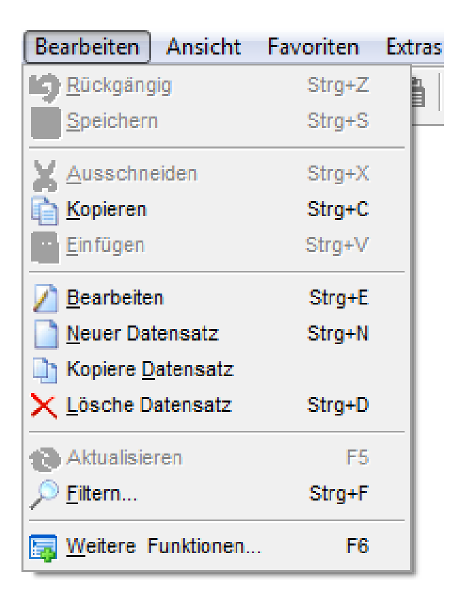


Mit einem Standard-*Datei/Öffnen*-Dialog wird die Komplexität von Menüs wesentlich reduziert. Der Benutzer öffnet Formulare immer durch einen einheitlichen Öffnen-Dialog. Standardmäßig wird der Öffnen-Dialog im Windows-XP-Stil am linken Bildschirmrand angezeigt.

VFX-Anwendungen bieten, dem *Office-Compatible*-Standard folgend, im Menü *Datei* eine Liste der zuletzt geöffneten Dateien an. Wie viele Dateien angezeigt werden, ist für jeden Benutzer in der Benutzerverwaltung individuell einstellbar.

Auch die *Datei/Beenden* Option entspricht dem *Office-Compatible*-Standard.

7.1.2. Menü: Bearbeiten



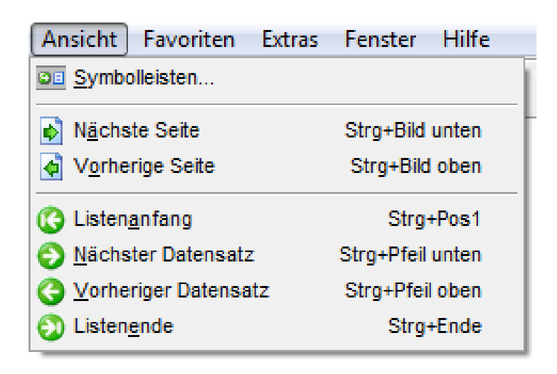
Hier befinden sich alle Funktionen zur Datenbearbeitung, die sich auf den aktuellen Datensatz beziehen sowie die Möglichkeit, die Dialoge für Filtern und weitere Funktionen aufzurufen. Je nach Status des Formulars

- Bearbeitungsmodus (oForm.*nFormStatus* = 1),
- Einfügemodus (oForm.*nFormStatus* = 2) oder
- Anzeigemodus (oForm.*nFormStatus* = 0)

sind einige der Optionen nicht verfügbar.

Um weitere Informationen zu erhalten, sehen Sie bitte im Kapitel *Das VFX Datenbearbeitungsformular* nach.

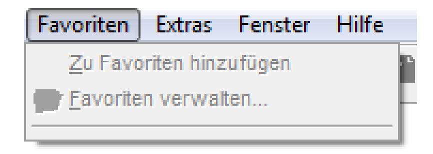
7.1.3. Menü: Ansicht



Hier können Sie den Symbolleisten-Dialog aufrufen, die Seite bei mehrseitigen Eingabeformularen wechseln, sowie den Datensatzzeiger bewegen.

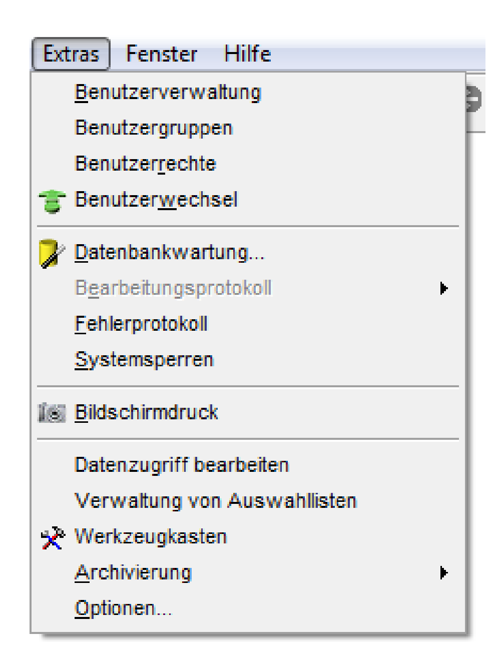
Um weitere Informationen zu erhalten, sehen Sie bitte im Kapitel *Das VFX Datenbearbeitungsformular* nach.

7.1.4. Menü: Favoriten



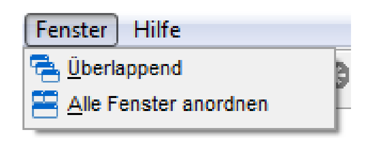
Dies ist das VFX-Favoriten-Menü. Mit der ersten Option wird der aktuelle Datensatz dem Favoriten-Menü hinzugefügt. Mit dem zweiten Eintrag werden die Favoriten verwaltet. Für alle verfügbaren Favoriten, gruppiert nach Formularen, werden Menüeinträge zur Laufzeit hinzugefügt.

7.1.5. Menü: Extras



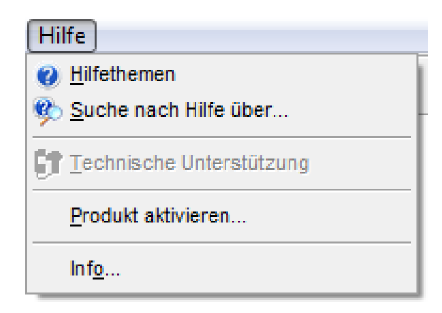
Um weitere Informationen zu den einzelnen Optionen zu erhalten, lesen Sie bitte in den Kapiteln Benutzerverwaltung, Benutzerrechte, Benutzerwechsel, Datenbankwartung, Bearbeitungsprotokoll und Fehlerprotokoll in diesem Handbuches nach.

7.1.6. Menü: Fenster



Falls Sie mehrere Fenster geöffnet haben, können Sie diese im Menü *Fenster* auswählen.

7.1.7. Menü: Hilfe



Das Hilfemenü bietet direkten Zugriff auf die Hilfedatei.

7.1.8. Standard-Symbolleiste

VFX-Anwendungen haben eine Standard-Symbolleiste, die Sie einfach um Ihre anwendungsspezifischen Schaltflächen erweitern können. Dadurch haben Benutzer einfachen Zugriff auf die Funktionen, die Ihre Anwendung bietet. Die VFX-Symbolleisten erscheinen im „Hot Tracking“ Layout.

Neu (Strg+N)	Anlegen eines neuen Datensatzes.
Kopiere Datensatz	Der angezeigte Datensatz wird in einen neuen Datensatz kopiert.
Öffnen (Strg+O)	Öffnet den Öffnen-Dialog am linken Bildschirmrand.
Speichern (Strg+S)	Speichern der Änderungen im aktiven Formular.
E-Mail	Versenden einer E-Mail aus der Berichtsausgabe aus dem aktiven Formular.
Drucken (Strg+P)	Drucken eines Berichts oder einer Liste aus dem aktiven Formular.
Seitenansicht	Anzeige der Druckvorschau eines Berichts oder einer Liste aus dem aktiven Formular.
Fax	Versenden eines Fax aus der Berichtsausgabe aus dem aktiven Formular.
Ausschneiden (Strg+X)	Entfernt die Markierung und überträgt sie in die Zwischenablage.
Kopieren (Strg+C)	Kopiert die Markierung in die Zwischenablage.
Einfügen (Strg+V)	Fügt den Inhalt der Zwischenablage ein.
Rückgängig (Strg+Z)	Macht die Änderungen in aktuellen Formular rückgängig.
Weitere Funktionen (F6)	Öffnet das Fenster mit weiteren Funktionen zum aktuellen Formular.
Bearbeitungsprotokoll	Öffnet das Formular mit dem Bearbeitungsprotokoll zum aktuellen Datensatz im aktiven Formular.
Bildschirminhalt drucken	Die aktuelle Bildschirmansicht wird gedruckt.

Bearbeiten (Strg+E)	Schaltet das aktive Formular in den Bearbeitungsmodus.
Löschen (Strg+D)	Löscht den aktuellen Datensatz im aktiven Formular.
Filtern (Strg+F)	Filtern der Daten im aktiven Formular nach einzugebenden Kriterien.
Anfang (Strg+Pos1)	Bewegt den Datensatzzeiger auf den Anfang der Tabelle oder Ansicht.
Rückwärts blättern (Strg+Pfeil oben)	Bewegt den Datensatzzeiger auf den vorherigen Datensatz der Tabelle oder Ansicht.
Vorwärts blättern (Strg+Pfeil unten)	Bewegt den Datensatzzeiger auf den nächsten Datensatz der Tabelle oder Ansicht.
Ende (Strg+Ende)	Bewegt den Datensatzzeiger auf das Ende der Tabelle oder Ansicht.
User	Beispiel für eine individuell zu verwendende Schaltfläche.
Refresh	Aktualisieren der Ansicht des aktiven Formulars nach der Eingabe von Parametern zur Datenselektion.
Hilfe (F1)	Aufruf der kontextsensitiven Hilfe.
Benutzerwechsel	Ermöglicht die Anmeldung eines anderen Benutzers während das Programm läuft.
Schließen (ESC)	Das aktive Formular wird geschlossen.

Neben dieser Standard-Symbolleiste bietet Ihnen VFX an, eine formularspezifische Symbolleiste zu definieren. Alles was Sie tun müssen, ist eine Symbolleisten-Klasse zu definieren und den Namen dieser Symbolleiste in der Formular-Eigenschaft *CToolbarClass* einzutragen. VFX erledigt alles Weitere für Sie automatisch.

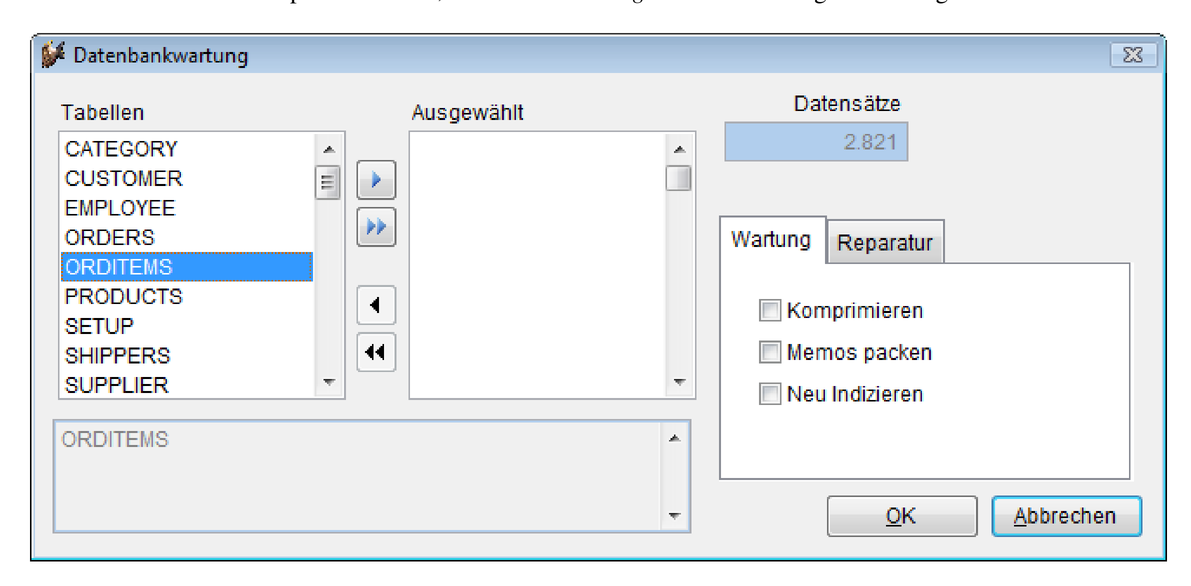
HINWEIS: Für eine ausführliche technische Beschreibung zur Benutzung von formularspezifischen Symbolleisten lesen Sie bitte in der VFX Technischen Referenz nach.

7.1.9. Abschließende Bemerkung zur Office-Kompatibilität

Je nach Art Ihrer Anwendung kann es erforderlich sein, vom Office-Compatible-Standard abzuweichen. Das VFX-Menü zeigt eine Alternative, die die meisten Bedürfnisse, aber nicht alle, von möglichen Anwendungen abdeckt. Es lohnt sich einige Zeit in den Aufbau des Menüs und der Symbolleisten zu investieren, die Sie in Ihren Anwendungen verwenden wollen.

7.2. Datenbankwartung

Durch Auswahl des Menüpunktes *Extras, Datenbankwartung* erscheint der folgende Dialog:



In diesem Dialog sehen Sie eine Liste mit allen in Ihrer Anwendung verfügbaren Tabellen. In einem einfach zu bedienenden VFX-Mover-Dialog können die Tabellen ausgewählt werden, die bearbeitet werden sollen.

Es kann aus einer der folgenden Optionen ausgewählt werden:

- Komprimieren (pack)
- Memos packen (pack memo)
- Neu indizieren (reindex)

Drücken Sie nach der Auswahl auf *OK,* um die gewünschte Datenbankwartung durchzuführen.

HINWEIS: Der hier verwendete Mover-Dialog ist ebenfalls eine VFX-Klasse und steht auch für Ihre eigenen Anwendungen zur Verfügung.

Zusätzlich zu den Datenbank-Wartungsmöglichkeiten aus bisherigen enthält VFX 9. ein neues Werkzeug zur Reparatur von defekten Datenbanken. Die Reparaturmöglichkeit von Datenbanken ist den Dialog *Datenbank-wartung* integriert.

Bei Bedarf können wahlweise ausgewählte Tabellen oder die gesamte Datenbank repariert werden. Wenn nur ausgewählte Tabellen repariert werden sollen, kann nur der Tabellenkopf repariert werden oder es werden defekte Datensätze gelöscht.

Zur Datenbankreparatur wird eine leere Datenbank benötigt, die die gleiche Struktur wie die beschädigte Datenbank hat. Vor der Erstellung einer ausführbaren Datei wird mithilfe von *Gendbc.prg* ein Programm erstellt, das diese Struktur zur Laufzeit herstellen kann. Das generierte Programm wird dem Projekt automatisch hinzugefügt.

Wenn *beschädigte Datensätze löschen* ausgewählt wird, werden alle Datensätze ohne Primärschlüssel oder mit doppeltem Primärschlüssel gelöscht.

7.3. Benutzerverwaltung

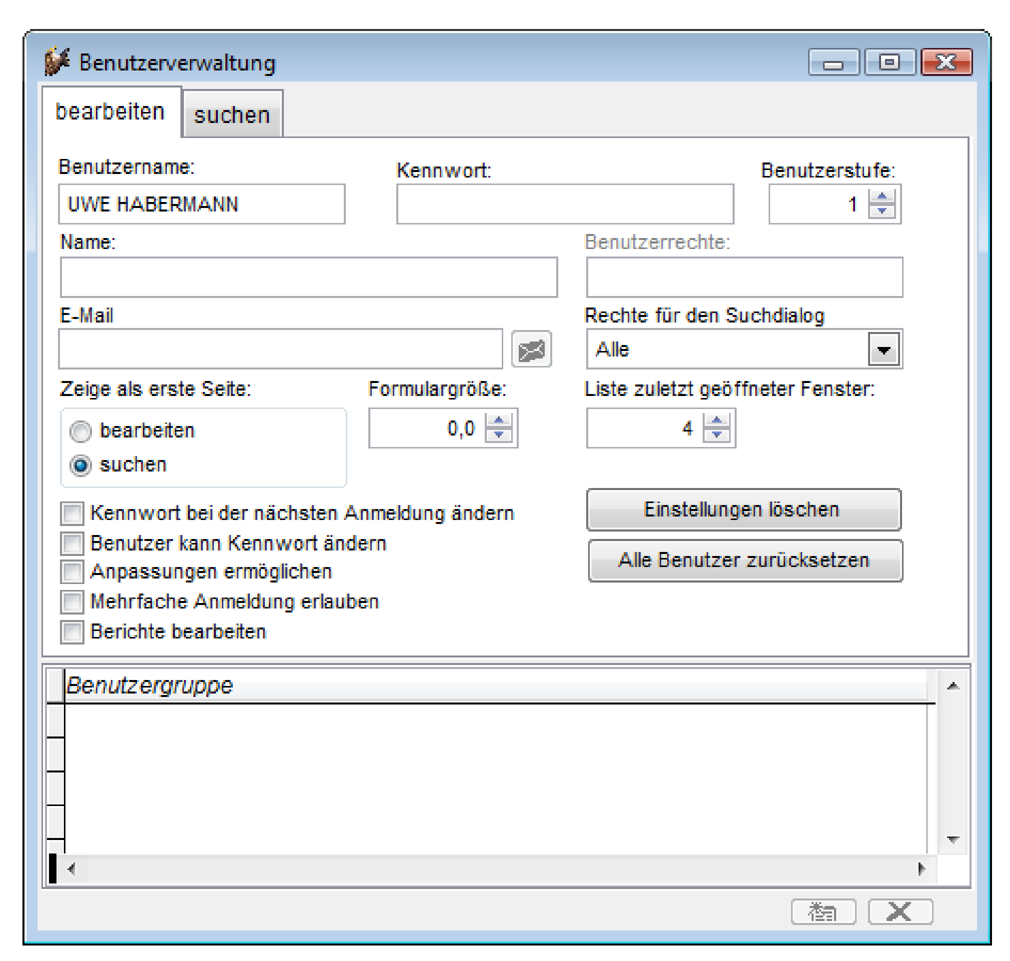
In jeder Mehrbenutzeranwendung sollte eine Benutzerverwaltung vorhanden sein. Als erstes muss festgelegt werden, wer zu Ihrer Anwendung Zugang hat. Dazu werden der Benutzername, das Kennwort und die Zugriffsrechte je Benutzer gespeichert.

Die Tabelle, in der die benutzerspezifischen Daten gespeichert sind, ist die freie Tabelle *Vfxusr.dbf/cdx*. Wenn Sie den Vorteil der langen Feldnamen nutzen möchten, können Sie diese Tabelle in Ihren Datenbank-Container einfügen.

Benutzer können ihre eigenen Daten in der VFX-Ressourcentabelle löschen wenn sie mit neuen Einstellungen weitermachen wollen oder wenn sie von einer großen Bildschirmauflösung zu einer kleineren wechseln wollen oder wenn sie mit ihren bisherigen Einstellungen nicht mehr zufrieden sind. In der Ressourcentabelle werden die Einstellungen für Formulargröße, Spaltenbreiten in Grids und Sortierfolgen in Grids und Auswahl-Grids gespeichert. Um die Daten in der VFX-Ressourcentabelle zu löschen, drücken Sie auf die Schaltfläche *Einstellungen löschen*.

Die Benutzerverwaltung wurde in VFX 11.0 stark erweitert. Der Administrator kann jetzt mit der Schaltfläche „*Alle Benutzer zurücksetzen*" die Ressourcen für alle Benutzer zurücksetzen.

Für jeden Benutzer kann der Administrator einstellen, dass das Kennwort bei der nächsten Anmeldung geändert werden muss. Der Administrator kann auch einstellen, dass ein Benutzer sein Kennwort nicht ändern kann.



Benutzer haben erweiterte Möglichkeiten ihre Umgebung anzupassen. Der Entwickler kann es Benutzern erlauben ihre Umgebungseinstellungen zu ändern, indem die Eigenschaft *lAllowUserCustomization* des Anwendungsobjekts auf .T. eingestellt wird.

```
goProgram.lAllowUserCustomization=.T.
```

Wenn diese Eigenschaft auf .T. eingestellt ist, kann der Administrator allen Benutzern erlauben die Umgebungseinstellungen zu ändern. Wenn diese Eigenschaft auf .F. eingestellt ist, ist das Kontrollkästchen *Anpassungen je Benutzer ermöglichen* für den Administrator nicht sichtbar und die Umgebungseinstellungen können in der Anwendung grundsätzlich nicht eingestellt werden.

Wenn der Administrator anderen Benutzern nicht erlaubt Umgebungseinstellungen anzupassen, gelten die Einstellungen des Administrators für alle Benutzer der Anwendung.

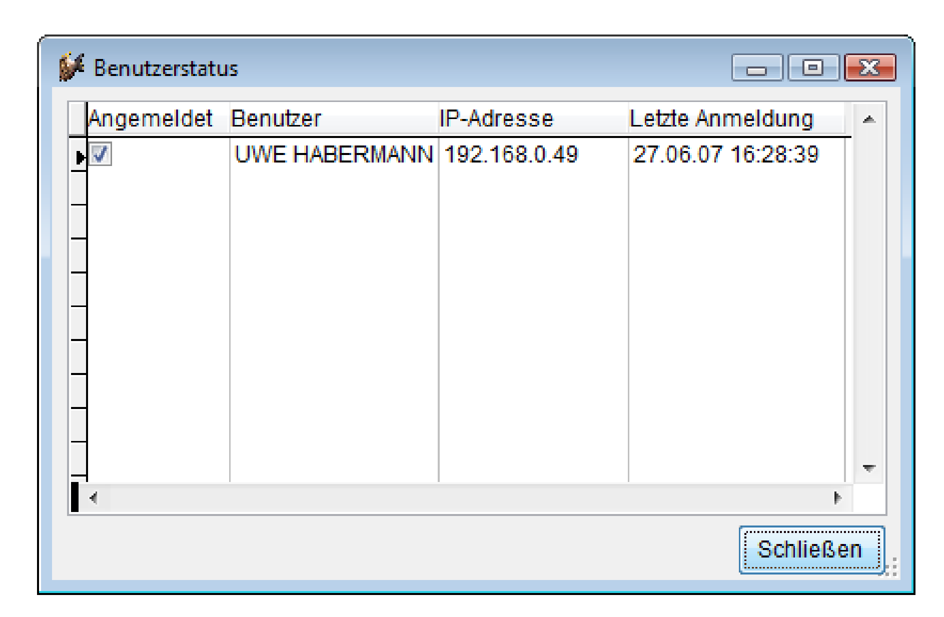
7.3.1. Zurzeit angemeldete Benutzer

VFX verwaltet zurzeit angemeldete Benutzer in einer Tabelle. Mit der Eigenschaft *lAllowMultipleLogin* des Anwendungsobjekts kann eingestellt werden, ob sich Benutzer mehrmals gleichzeitig an der Anwendung anmelden können. Wenn der Wert dieser Eigenschaft auf .T. eingestellt ist, können sich Benutzer mehrmals anmelden. Der Standardwert ist .T.

```
goProgram.lAllowMultipleLogin=.T.
```

Für jeden Benutzer wird die IP-Adresse des Arbeitsplatzes gespeichert, von dem aus er sich angemeldet hat. Wenn sich ein Benutzer abmeldet, wird die IP-Adresse gelöscht.

Benutzer mit Administratorrechten können über den Menüpunkt *Extras, Benutzerstatus* sehen, welche Benutzer zurzeit angemeldet sind. Es werden die IP-Adresse und die Anmeldezeit angezeigt. Die Spalte Anmeldezeit zeigt in jedem Fall das Datum und die Zeit der letzten Anmeldung, auch wenn der Benutzer zurzeit nicht angemeldet ist.



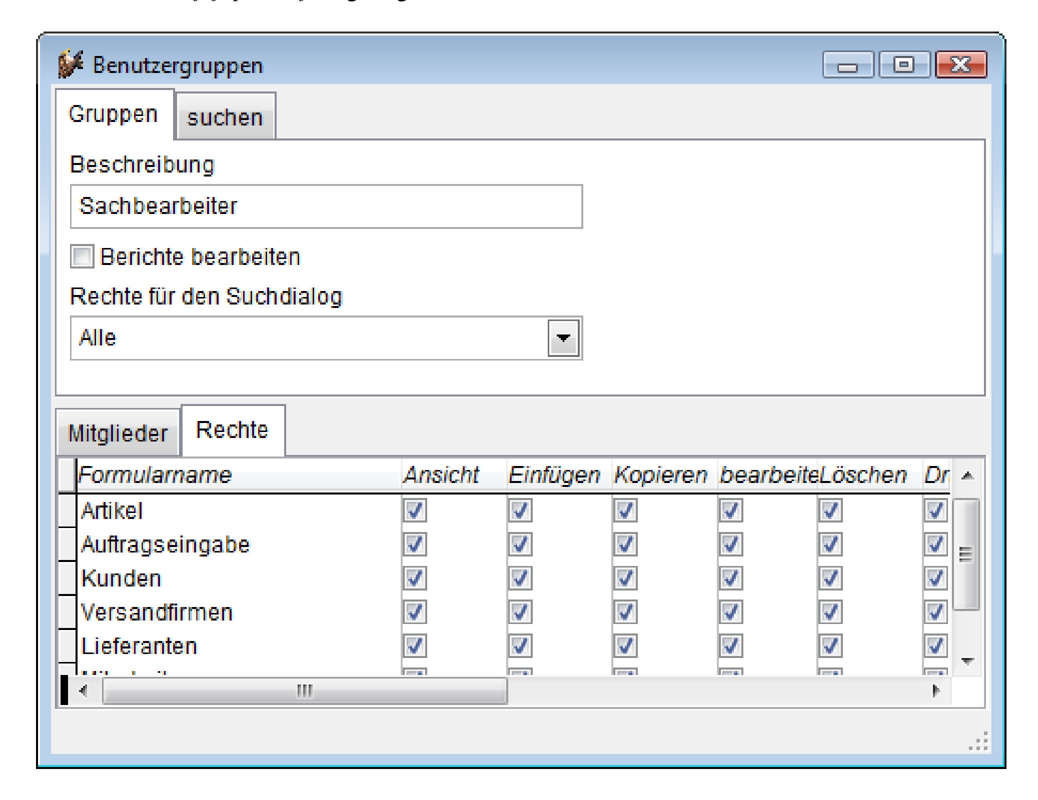
Wenn mit einer VFP-Datenbank gearbeitet wird, ist der Datensatz für den angemeldeten Benutzer ständig gesperrt. Im Falle einer Verbindungsunterbrechung oder eines Programmabbruchs, wird die Satzsperre automatisch aufgehoben. Der Benutzer kann sich erneut anmelden, ohne dass eine Mehrfachanmeldung festgestellt wird.

Wenn die VFX-Tabellen in einer SQL Server-Datenbank gespeichert sind, wird die System Prozess ID verwendet, um den an den SQL Server angemeldeten Benutzer zu identifizieren. Die aktuelle *SPID* wird in der *Vfxusr*-Tabelle gespeichert. Bei einer versuchten zweiten Anmeldung kann so festgestellt werden, ob der Benutzer bereits angemeldet ist. Wenn eine mehrfache Anmeldung nicht erlaubt ist, wird der Benutzer zurückgewiesen.

7.4. Benutzergruppen

Zusätzlich zu den bisherigen Möglichkeiten zur Vergabe von Benutzerrechten, können jetzt Benutzergruppen angelegt werden. Benutzer können Mitglied von einer oder mehreren Benutzergruppen sein. Benutzergruppen können Rechte zugewiesen werden. Wenn ein Benutzer Mitglied von mehreren Benutzergruppen ist, erhält er die Rechte von allen Benutzergruppen.

Benutzer mit Administratorrechten (Benutzerstufe 1) können Benutzergruppen anlegen und jeder Gruppe für jedes Formular individuelle Rechte zuweisen. Benutzerrechte können für alle Formulare eingestellt werden, die in der Tabelle *Vfxfopen.dbf* eingetragen sind.

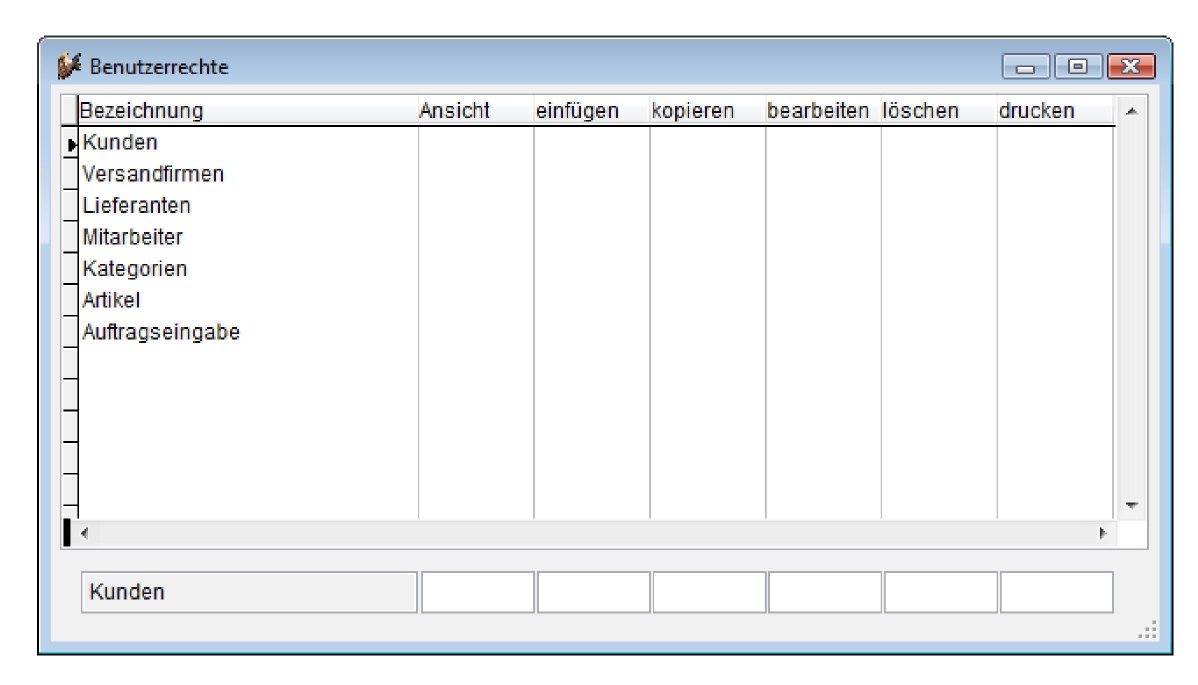


Zur Laufzeit wird ein globales Objekt *goUserRights* instanziiert. Dieses Objekt enthält Child-Objekte für jedes Formular der Anwendung. Die Namen dieser Objekte entsprechen den Namen der Formulare. Jedes dieser Objekte besitzt die Eigenschaften *deletepermit*, *editpermit*, *newpermit*, *printpermit* und *viewpermit*.

Die Eigenschaften des Objekts *goUserRights* sehen zur Laufzeit so aus:

Name	Value	Type
goUserRights	(Object)	O
frminvoices	(Object)	O
deletepermit	.F.	L
editpermit	.T.	L
newpermit	.T.	L
printpermit	.T.	L
viewpermit	.T.	L
frmorders	(Object)	O
deletepermit	.T.	L
editpermit	.T.	L
newpermit	.T.	L
printpermit	.T.	L
viewpermit	.T.	L

Wenn einem Benutzer keiner Benutzergruppe zugeordnet ist, gilt die Benutzerstufe wie in früheren VFX-Versionen.



Der Administrator hat die Benutzerstufe 1 und damit alle Rechte. Ein Benutzer, der die Benutzerstufe 99 hat, hat die wenigsten Rechte. Im Formular Benutzerrechte kann für jedes Formular festgelegt werden, welche Benutzerstufe erforderlich ist um das Formular anzeigen zu können, um neue Datensätze erfassen zu können, um vorhandenen Datensätze bearbeiten zu können und um Datensätze löschen zu können.

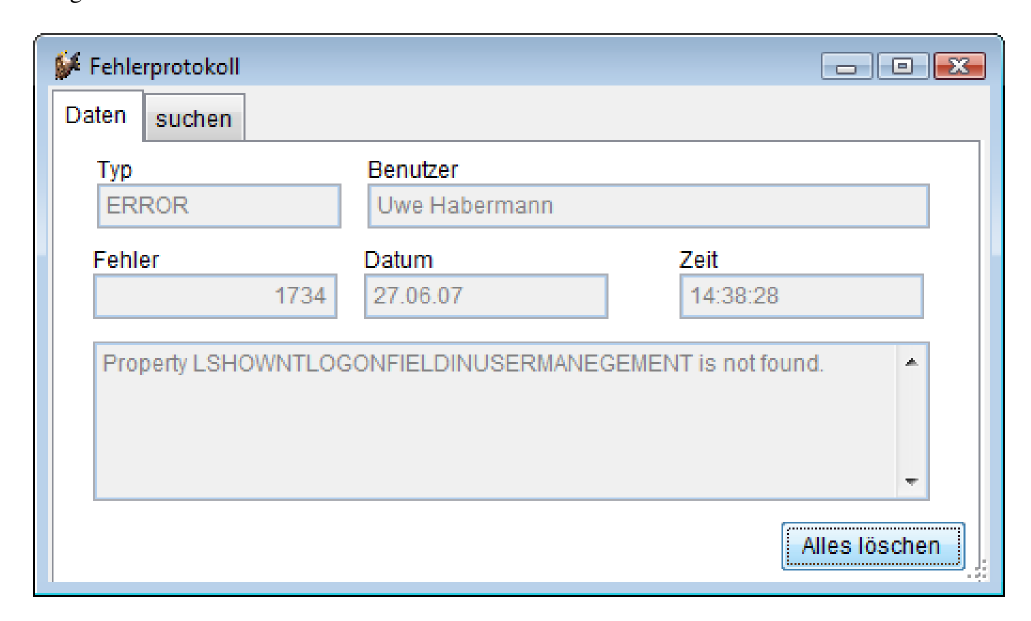
ANMERKUNG: Benutzer können nicht die Daten von anderen Benutzern ändern, wenn diese eine höhere Sicherheitsstufe haben. Sicherheitsstufen starten mit 1 (Administrator) und enden mit 99 (niedrigste Sicherheitsstufe). Zusätzlich können Sie eine Zugriffszeichenfolge für die weitere Anpassung an Ihre Bedürfnisse festlegen. Für weitere Sicherheitsaspekte, besonders für alle VFX Formular-Sicherheitseigenschaften, lesen Sie bitte in der VFX Technischen Referenz nach.

Wenn ein Benutzer nicht das Recht hat ein Formular anzuzeigen, wird das betreffende Formular nicht instanziiert. Solange im Dialog Benutzerrechte keine Benutzerstufen eingetragen sind, gelten die Einstellungen, die mit dem VFX – Form Wizard in den Formular-Eigenschaften *lcaninsert*, *lcancopy*, *lcanedit* und *lcandelete* hinterlegt sind.

7.5. Fehlerprotokoll

VFX protokolliert alle Laufzeitfehler automatisch. Die Tabelle mit den Fehlermeldungen ist die freie Tabelle *Vfxlog.dbf/cdx*.

Das Bearbeitungsformular, basierend auf der Klasse *CDataFormPage*, wird automatisch vom VFX Anwendungs-Assistenten vorbereitet.



Der Administrator kann das Fehlerprotokoll mit der Schaltfläche *Alles löschen* löschen.

ANMERKUNG: Für weitere Informationen lesen Sie bitte in der VFX Technischen Referenz nach.

7.6. Fehlerbehandlung

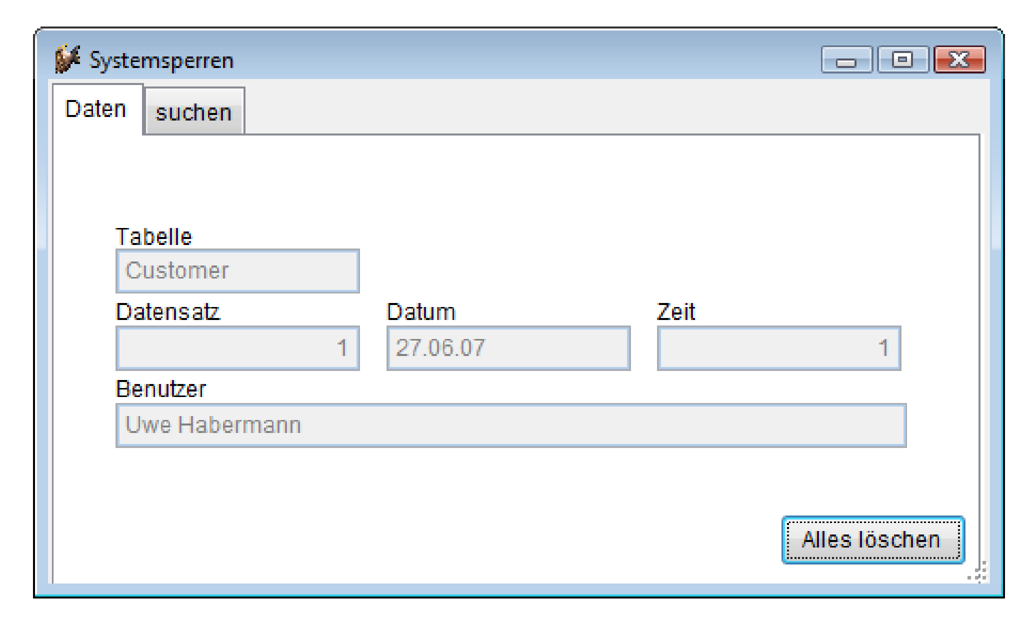
In VFX 11.0 ist eine erweiterte Behandlung von Laufzeitfehlern implementiert. Das Laufzeitfehlerprotokoll kann vom Kunden jetzt per E-Mail an den Entwickler gesendet werden. Der Kunde wird über den Inhalt des Fehlerberichts informiert. Der Versand des Fehlerberichts als E-Mail an den Entwickler ist der schnellste Weg Probleme in einer Anwendung zu lokalisieren und zu beseitigen. Die E-Mailadresse des Entwicklers wird der Eigenschaft *goProgram.csupportemail* zugewiesen. Der Wert dieser Eigenschaft kann mit dem VFX – Application Builder bearbeitet werden.

7.7. Systemsperren

In viel benutzten Mehrbenutzerumgebungen kann eine Meldung wie *“Datensatz durch anderen Benutzer gesperrt”* unter Umständen nicht ausreichen. Für solche Fälle stellt VFX eine System-Sperrentabelle zur Verfügung. In dieser Tabelle wird gespeichert, welcher Benutzer seit wann welchen oder welche Datensätze in Benutzung hat. (Siehe die Funktionen *XLock()* sowie *XUnlock()* in der Technischen Referenz unter *Funktionen*).

Die Systemsperrentabelle, in der alle Sperren mit VFX-Funktionsaufrufen gespeichert werden, ist die freie Tabelle *Vfxlock.dbf/cdx*.

Das Bearbeitungsformular basiert auf der VFX-Klasse *CDataFormPage* und wird automatisch durch den VFX Anwendungs-Assistenten vorbereitet.

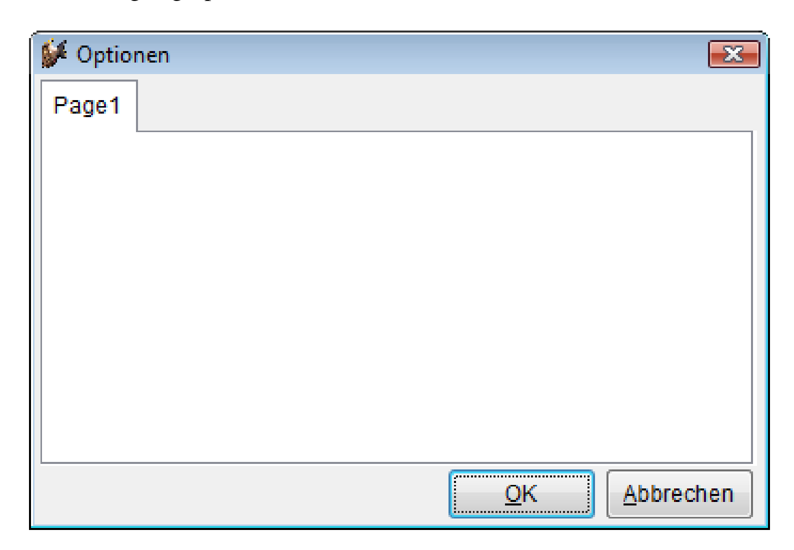


Der Administrator kann die Systemsperren mit der Schaltfläche *Alles löschen* löschen.

ANMERKUNG: Für weitere Informationen lesen Sie bitte in der VFX Technischen Referenz nach.

7.8. Optionen

Im Gegensatz zu den benutzerspezifischen Einstellungen werden in der Tabelle *Vfxsys.dbf* die systemspezifischen Einstellungen gespeichert.



Das oben abgebildete Formular ist eine Vorlage, die für die eigenen Optionen verwendet werden kann.

Der VFX Anwendungs-Assistent erstellt das Formular *Vfxsys.scx* für Sie in einer gebrauchsfertigen Form. Dieses Formular basiert auf der Klasse *CSystemDialog*. Alles was Sie noch tun müssen ist, die gewünschten Felder in der *Vfxsys.dbf*-Tabelle anzulegen. Die entsprechenden Steuerelemente auf dem Formular bekommen als Controlsource eine Referenz auf eine Eigenschaft des Objekts *goSystem*.

Hier wird für jedes Feld aus der Tabelle *Vfxsys.dbf* eine Eigenschaft des Objekts *goSystem* angelegt. VFX übernimmt vollautomatisch das Speichern und Wiederherstellen dieser Werte falls diese aus dem Optionen-Dialog heraus verändert werden.

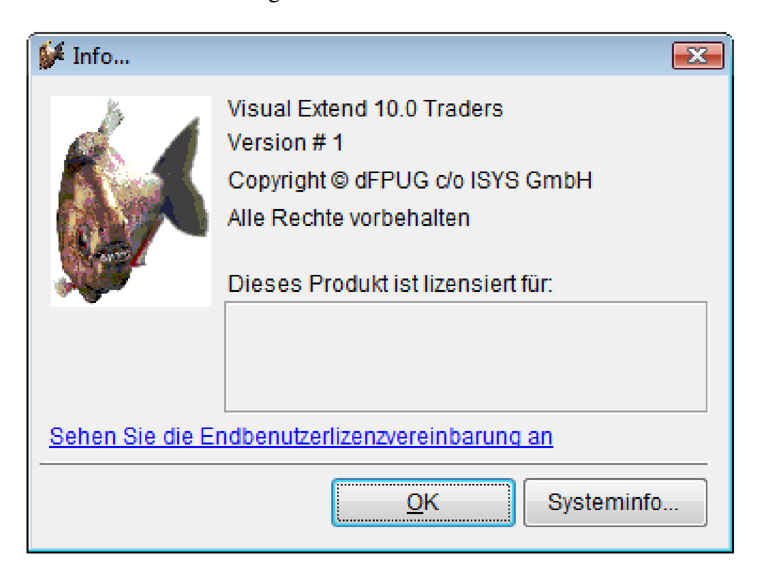
Wenn Sie ein Feld mit dem Namen *Test* in der Tabelle *Vfxsys.dbf* haben, wird eine Eigenschaft mit dem Namen *Test* und dem Wert aus dem Feld *Test* der *Vfxsys.dbf*-Tabelle angelegt. Falls diese Variable verändert wird, wird beim Verlassen des Optionen-Dialogs dieser Wert wieder zurück in das Feld *Test* der Tabelle *Vfxsys.dbf* geschrieben.

Auf diese Weise ist es sehr einfach, systemspezifische Einstellungen zu speichern und wiederherzustellen. Probieren Sie es!

7.9. Infodialog

Der VFX-Anwendungs-Assistent erstellt einen Infodialog, der auf der Klasse *CAboutDialog* basiert.

Sie finden den Infodialog im Menü Hilfe.



Um diesen Dialog Ihren Bedürfnissen anzupassen, steht Ihnen die Include-Datei *Usertxt.h* zur Verfügung:

```
…
#define CAP_APPLICATION_TITLE                    "VFX 11.00 Build 0000 Test Application"
#define CAP_LBLCOPYRIGHTINFORMATION              "Copyright © dFPUG c/o ISYS GmbH"
#define CAP_LBLTHISPRODUCTISLICENSEDTO           "This product is licensed to:"
#define CAP_LBLTRADEMARKINFORMATION              "Trademark Information"
#define CAP_LBLVERSION                           "Version "
#define CAP_LBLYOURAPPLICATIONNAME               "VFX Test Application"
…
```

HINWEIS: Wenn Sie Änderungen in dieser Include-Datei machen, müssen Sie das Formular *Vfxabout.scx* vor dem Start Ihrer Anwendung öffnen und speichern oder kompilieren, sonst werden die Änderungen in der Include-Datei nicht übernommen.

8. Die VFX Builder

Die VFX-Builder unterstützen den Entwickler bei der Erstellung und Bearbeitung Formularen, Grids und Auswahlfeldern.

Formulare manuell zu erstellen kann viel Zeit beanspruchen, insbesondere dann, wenn Sie viele Formulare mit vielen Feldern anzeigen möchten. Stellt man sich zum Beispiel ein Formular mit 20 Feldern vor, so hat man bereits 40 Steuerelemente, allein für die Dateneingabefelder: 20 Textfelder oder andere Steuerelemente und 20 Bezeichnungen. Wenn Sie Klassenbibliotheken verwenden, müssen die gewünschten Steuerelemente per drag & drop auf das Formular ziehen. Mit den VFX-Formular-Buildern ist diese Aufgabe sehr schnell und einfach durchführbar.

Ein weiterer großer Vorteil der VFX-Formular-Builder ist die Widerverwendbarkeit. Das bedeutet, dass Sie Änderungen, die Sie in Ihrer Datenbank gemacht haben, einfach in das bestehende Formular übernehmen können, indem Sie den VFX-Formular-Builder aufrufen und das Kontrollkästchen *Use DBC Definitions* auswählen. Auch Seiten zu einem Seitenrahmen hinzuzufügen oder Änderungen an den Spalten eines Grids sind sehr einfach dank der Widerverwendbarkeit der VFX-Formular-Builder.

Bitte lesen Sie den Abschnitt *Formularbedienung* später in diesem Handbuch, um eine Vorstellung von der Bedienung eines von VFX erzeugten Standard-Bearbeitungsformulars zu bekommen.

Zuerst müssen Sie die Datenbank für Ihre Anwendung erstellen. Legen Sie Ihre Tabellen, Felder und Indexschlüssel an.

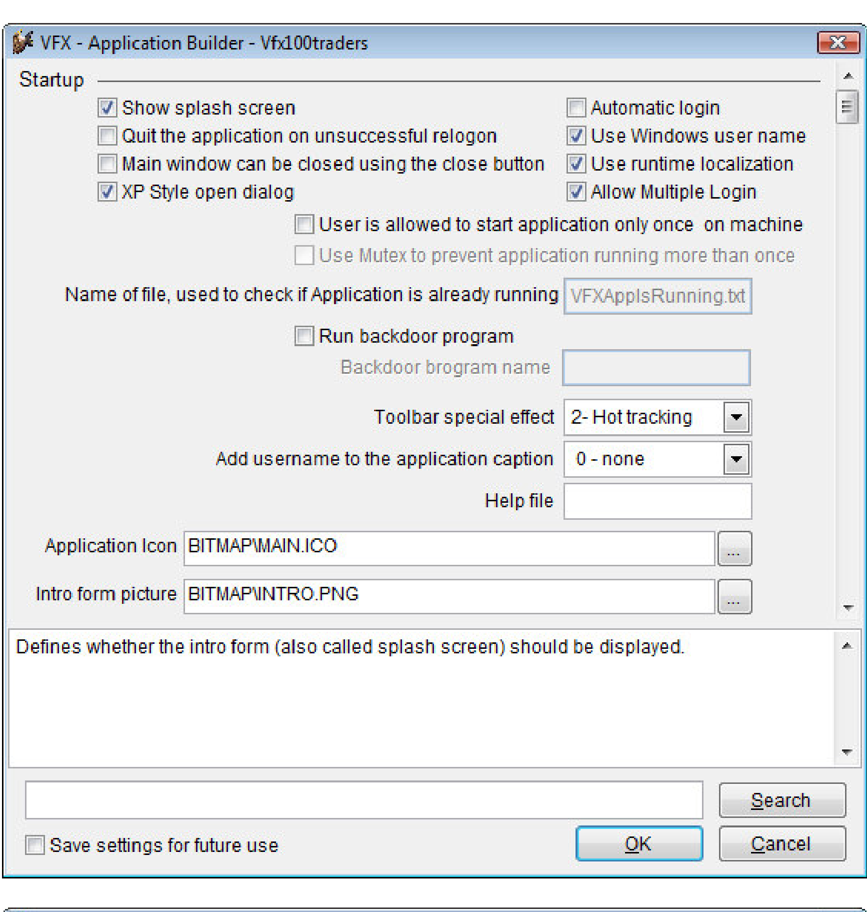
ANMERKUNG: Wenn Sie die Daten für Überschriften, Formate, Eingabeformulare und Bibliothek für Anzeige im Datenbank-Container speichern, werden diese automatisch von den VFX-Formular-Buildern und vom VFX-Grid-Builder verwendet.

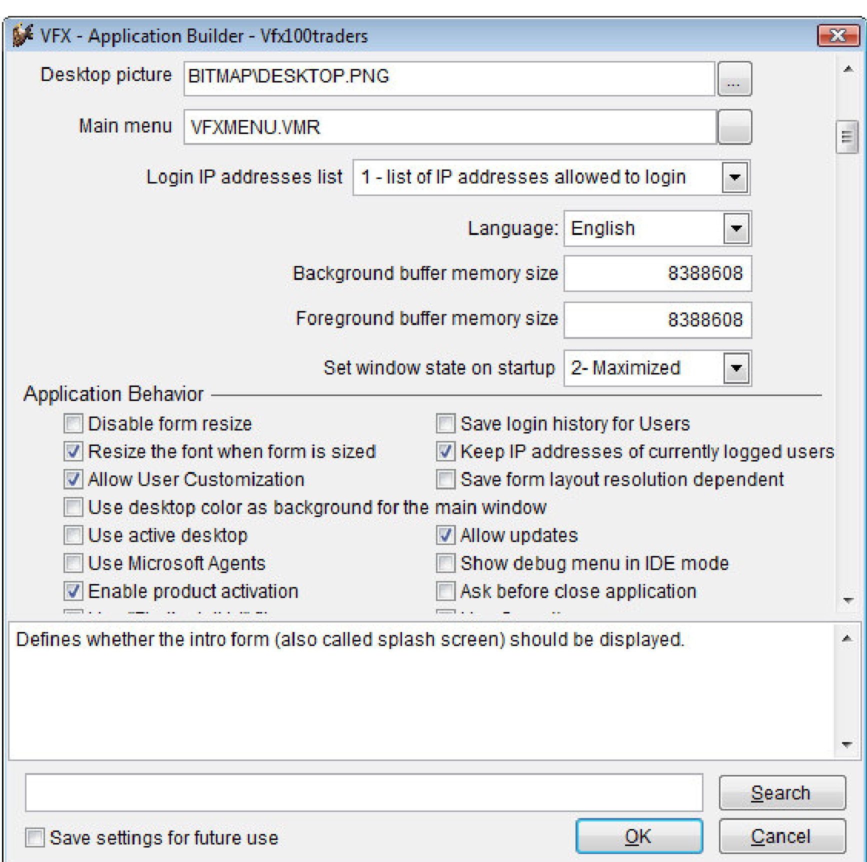
Wie wir bereits gesehen haben, geschieht das Erstellen neuer Projekte mit dem VFX – Application Wizard. Dieser Wizard kann über eine Schaltfläche in der VFX Task Pane oder über das VFX 11.0-Menü gestartet werden.

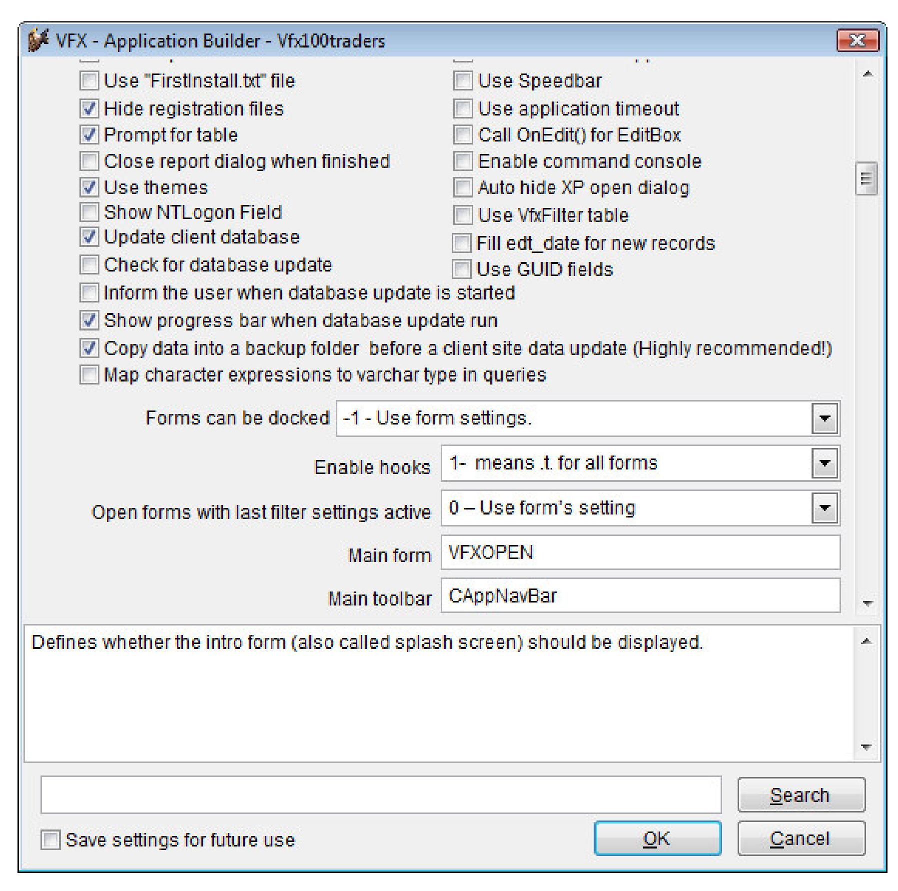
8.1. VFX – Application Builder

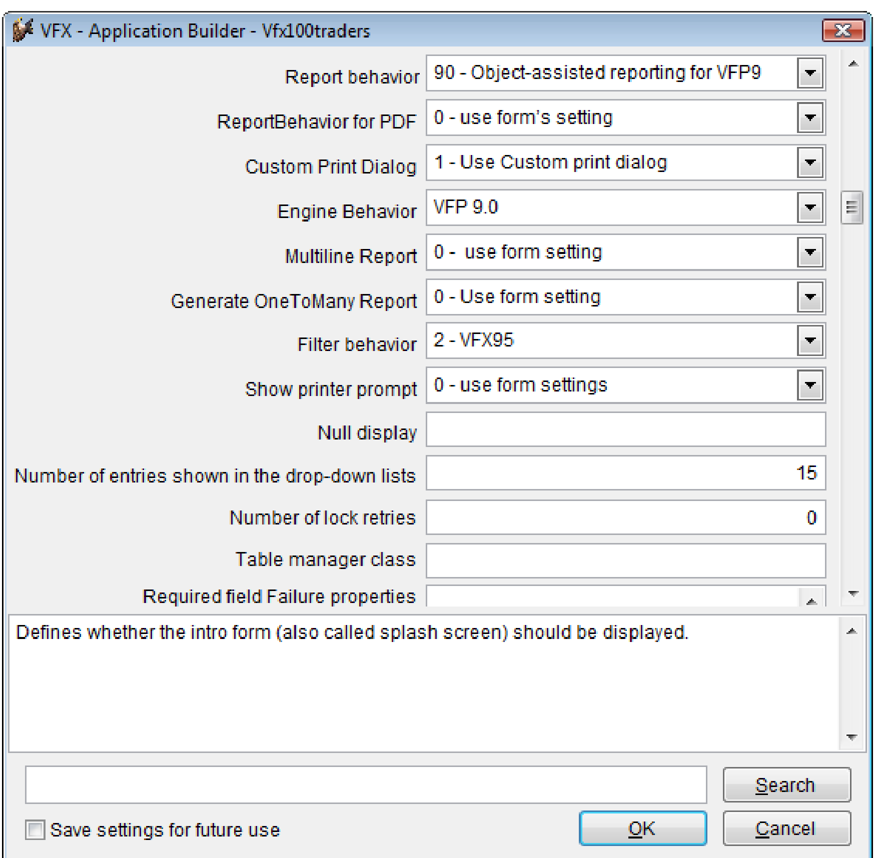
Dieser Dialog kann jederzeit über den Menüpunkt *Project, Application Builder* aufgerufen werden, um Einstellungen des Anwendungsobjekts zu ändern.

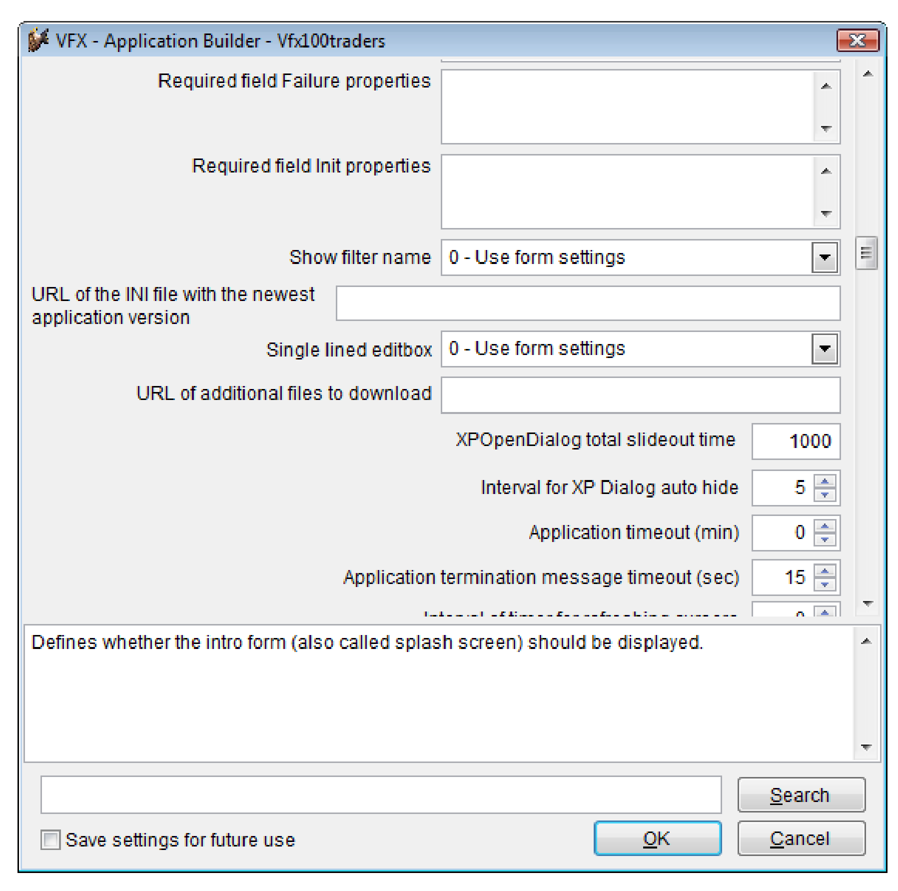
ANMERKUNG: Die mit dem VFX – Application Builder gemachten Einstellungen werden für das nächste neue Projekt übernommen.

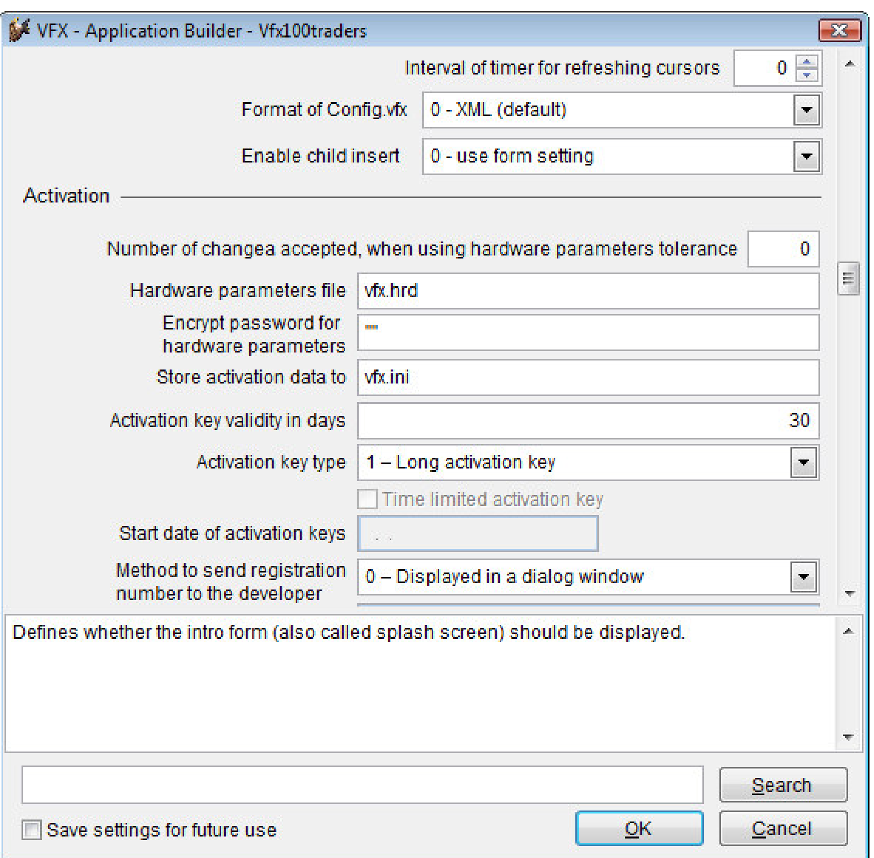

VFX - Application Builder - Vfx100traders
Startup
Show splash screen
Quit the application on unsuccessful relogon
Main window can be closed using the close button
XP Style open dialog
Automatic login
Use Windows user name
Use runtime localization
Allow Multiple Login
User is allowed to start application only once on machine
Use Mutex to prevent application running more than once
Name of file, used to check if Application is already running
VFXAppIsRunning.txt
Run backdoor program
Backdoor brogram name
Toolbar special effect
2- Hot tracking
Add username to the application caption
0 - none
Help file
Application Icon
BITMAP\MAIN.ICO
Intro form picture
BITMAP\INTRO.PNG
Defines whether the intro form (also called splash screen) should be displayed.
Search
Save settings for future use
OK
Cancel

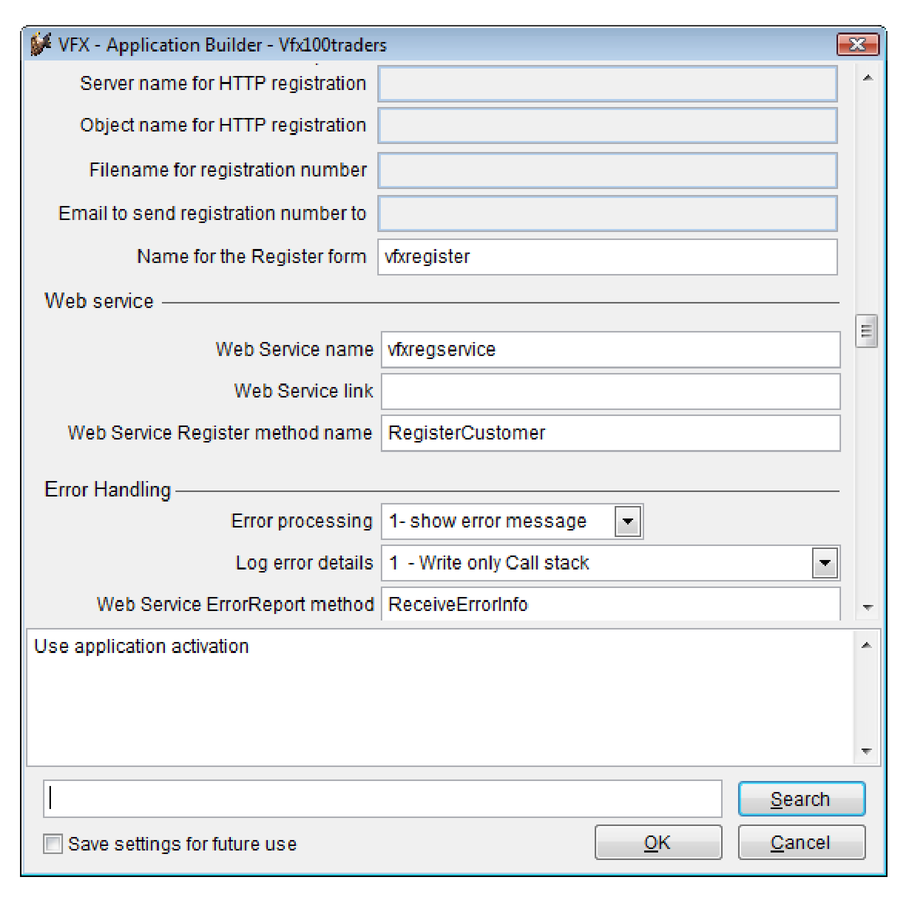

VFX - Application Builder - Vfx100traders
Desktop picture
BITMAP\DESKTOP.PNG
Main menu
VFXMENU.VMR
Login IP addresses list
1 - list of IP addresses allowed to login
Language:
English
Background buffer memory size
8388608
Foreground buffer memory size
8388608
Set window state on startup
2- Maximized
Application Behavior
Disable form resize
Resize the font when form is sized
Allow User Customization
Use desktop color as background for the main window
Use active desktop
Use Microsoft Agents
Enable product activation
Save login history for Users
Keep IP addresses of currently logged users
Save form layout resolution dependent
Allow updates
Show debug menu in IDE mode
Ask before close application
Defines whether the intro form (also called splash screen) should be displayed.
Search
Save settings for future use
OK
Cancel

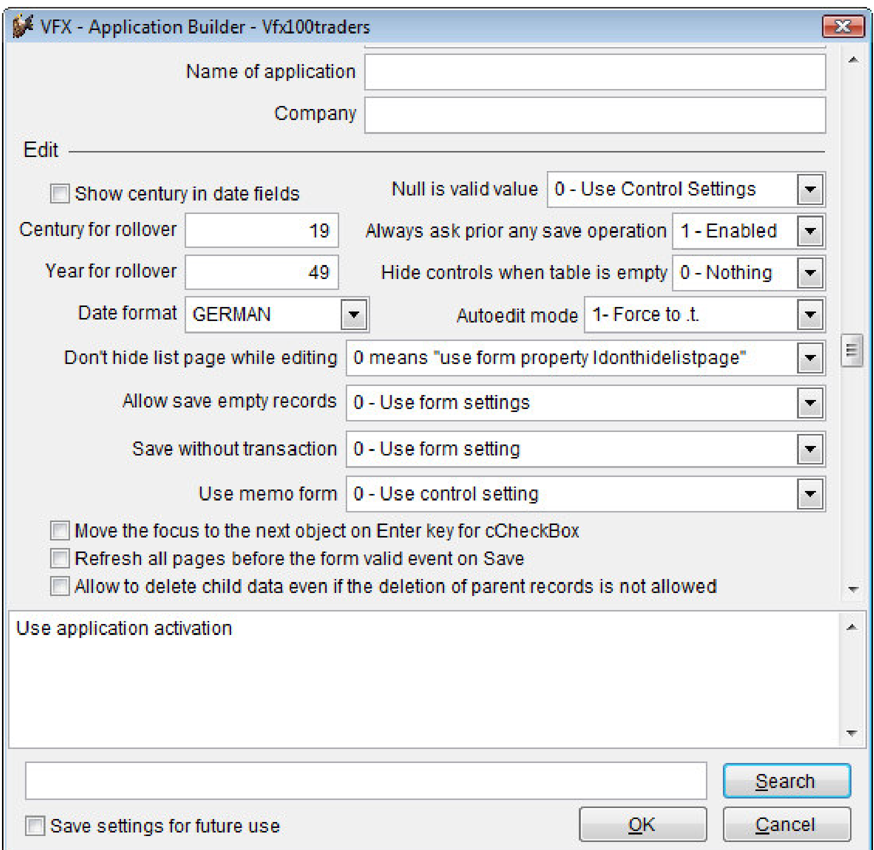

VFX - Application Builder - Vfx100traders
Use "FirstInstall.txt" file
Hide registration files
Prompt for table
Close report dialog when finished
Use themes
Show NTLogon Field
Update client database
Check for database update
Inform the user when database update is started
Show progress bar when database update run
Copy data into a backup folder before a client site data update (Highly recommended!)
Map character expressions to varchar type in queries
Use Speedbar
Use application timeout
Call OnEdit() for EditBox
Enable command console
Auto hide XP open dialog
Use VfxFilter table
Fill edt_date for new records
Use GUID fields
Forms can be docked
-1 - Use form settings.
Enable hooks
1- means .t. for all forms
Open forms with last filter settings active
0 – Use form's setting
Main form
VFXOPEN
Main toolbar
CAppNavBar
Defines whether the intro form (also called splash screen) should be displayed.
Search
Save settings for future use
OK
Cancel

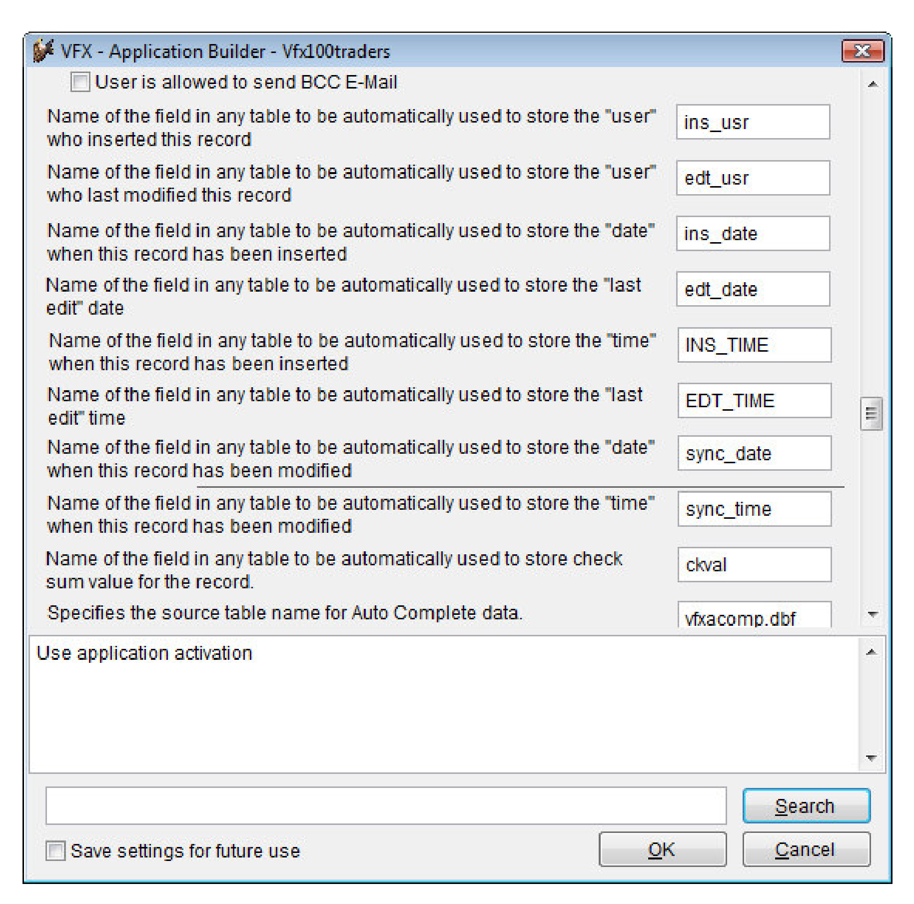

VFX - Application Builder - Vfx100traders
Report behavior
90 - Object-assisted reporting for VFP9
ReportBehavior for PDF
0 - use form's setting
Custom Print Dialog
1 - Use Custom print dialog
Engine Behavior
VFP 9.0
Multiline Report
0 - use form setting
Generate OneToMany Report
0 - Use form setting
Filter behavior
2 - VFX95
Show printer prompt
0 - use form settings
Null display
Number of entries shown in the drop-down lists
15
Number of lock retries
0
Table manager class
Required field Failure properties
Defines whether the intro form (also called splash screen) should be displayed.
Search
Save settings for future use
OK
Cancel

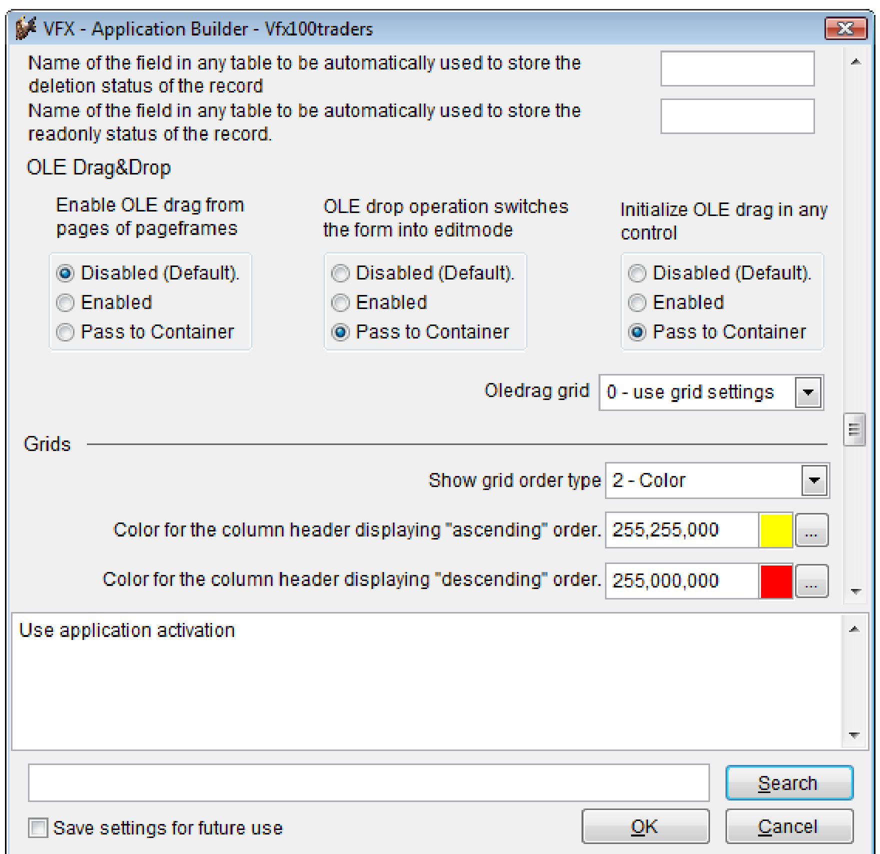

VFX - Application Builder - Vfx100traders
Required field Failure properties
Required field Init properties
Show filter name
0 - Use form settings
URL of the INI file with the newest application version
Single lined editbox
0 - Use form settings
URL of additional files to download
XPOpenDialog total slideout time
1000
Interval for XP Dialog auto hide
5
Application timeout (min)
0
Application termination message timeout (sec)
15
Defines whether the intro form (also called splash screen) should be displayed.
Search
Save settings for future use
OK
Cancel

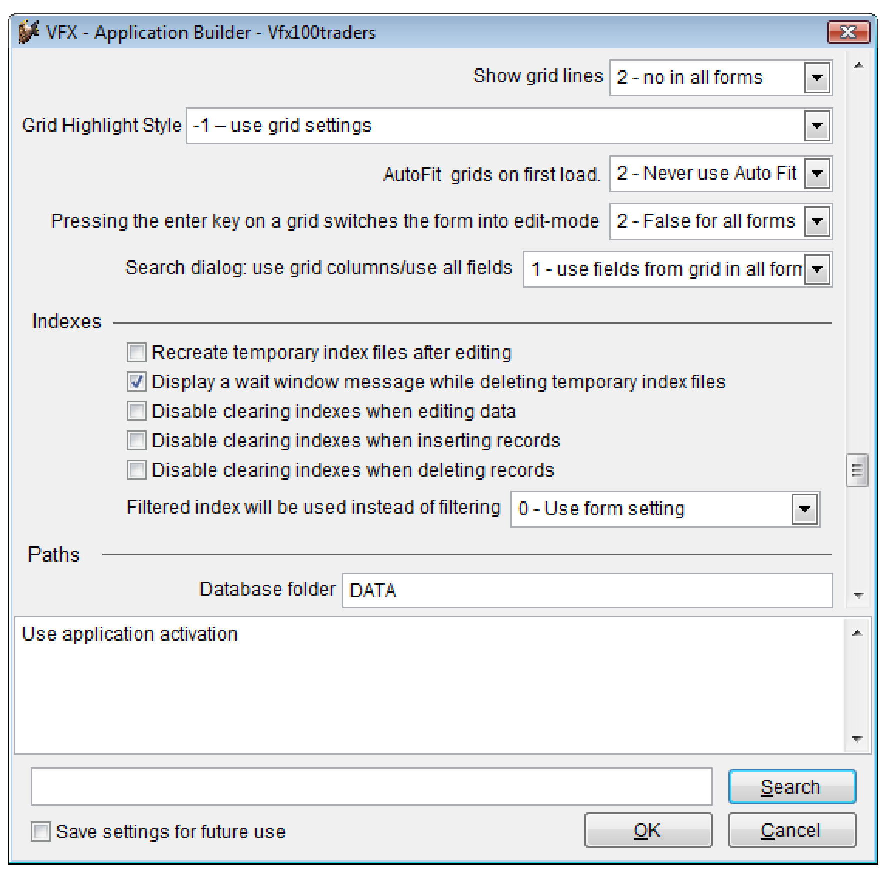

VFX - Application Builder - Vfx100traders
Interval of timer for refreshing cursors
0
Format of Config.vfx
0 - XML (default)
Enable child insert
0 - use form setting
Activation
Number of changea accepted, when using hardware parameters tolerance
0
Hardware parameters file
vfx.hrd
Encrypt password for hardware parameters
Store activation data to
vfx.ini
Activation key validity in days
30
Activation key type
1 – Long activation key
Time limited activation key
Start date of activation keys
Method to send registration number to the developer
0 – Displayed in a dialog window
Defines whether the intro form (also called splash screen) should be displayed.
Search
Save settings for future use
OK
Cancel

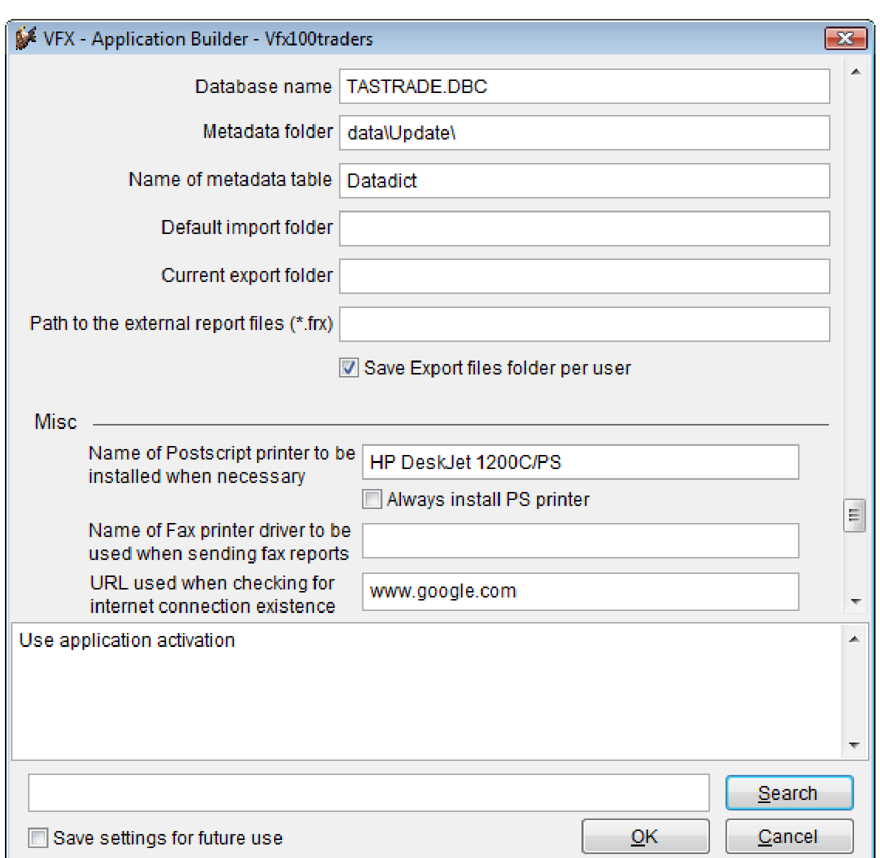

VFX - Application Builder - Vfx100traders
Server name for HTTP registration
Object name for HTTP registration
Filename for registration number
Email to send registration number to
Name for the Register form
vfxregister
Web service
Web Service name
vfxregservice
Web Service link
Web Service Register method name
RegisterCustomer
Error Handling
Error processing
1- show error message
Log error details
1 - Write only Call stack
Web Service ErrorReport method
ReceiveErrorInfo
Use application activation
Search
Save settings for future use
OK
Cancel

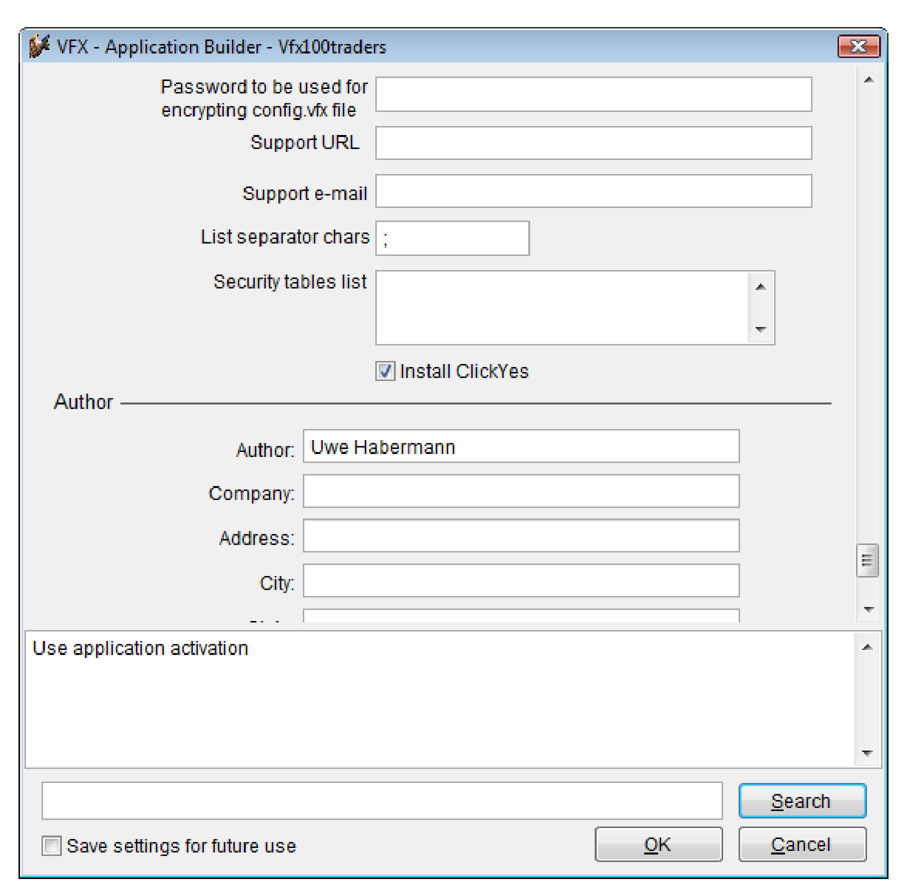

VFX - Application Builder - Vfx100traders
Name of application
Company
Edit
Show century in date fields
Null is valid value
0 - Use Control Settings
Century for rollover
19
Always ask prior any save operation
1 - Enabled
Year for rollover
49
Hide controls when table is empty
0 - Nothing
Date format
GERMAN
Autoedit mode
1- Force to .t.
Don't hide list page while editing
0 means "use form property ldonthidelistpage"
Allow save empty records
0 - Use form settings
Save without transaction
0 - Use form setting
Use memo form
0 - Use control setting
Move the focus to the next object on Enter key for cCheckBox
Refresh all pages before the form valid event on Save
Allow to delete child data even if the deletion of parent records is not allowed
Use application activation
Search
Save settings for future use
OK
Cancel

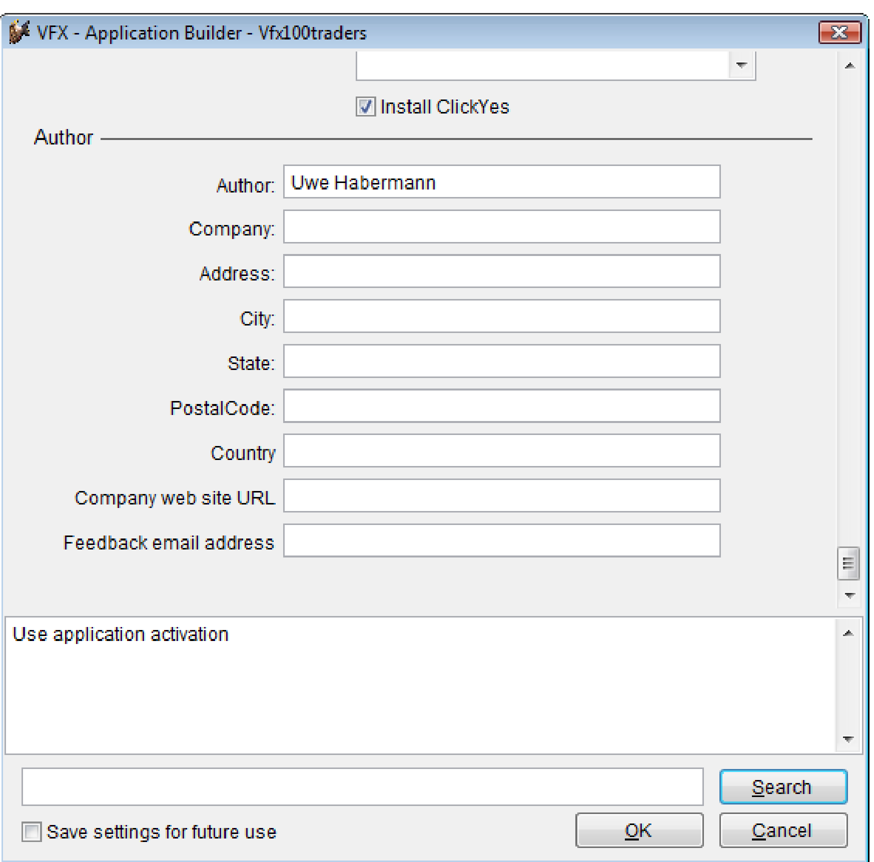

VFX - Application Builder - Vfx100traders
User is allowed to send BCC E-Mail
Name of the field in any table to be automatically used to store the "user" who inserted this record
ins_usr
Name of the field in any table to be automatically used to store the "user" who last modified this record
edt_usr
Name of the field in any table to be automatically used to store the "date" when this record has been inserted
ins_date
Name of the field in any table to be automatically used to store the "last edit" date
edt_date
Name of the field in any table to be automatically used to store the "time" when this record has been inserted
INS_TIME
Name of the field in any table to be automatically used to store the "last edit" time
EDT_TIME
Name of the field in any table to be automatically used to store the "date" when this record has been modified
sync_date
Name of the field in any table to be automatically used to store the "time" when this record has been modified
sync_time
Name of the field in any table to be automatically used to store check sum value for the record.
ckval
Specifies the source table name for Auto Complete data.
vfxacomp.dbf
Use application activation
Search
Save settings for future use
OK
Cancel


VFX - Application Builder - Vfx100traders
Name of the field in any table to be automatically used to store the deletion status of the record
Name of the field in any table to be automatically used to store the readonly status of the record.
OLE Drag&Drop
Enable OLE drag from pages of pageframes
OLE drop operation switches the form into editmode
Initialize OLE drag in any control
Disabled (Default).
Enabled
Pass to Container
Disabled (Default).
Enabled
Pass to Container
Disabled (Default).
Enabled
Pass to Container
Oledrag grid
0 - use grid settings
Grids
Show grid order type
2 - Color
Color for the column header displaying "ascending" order.
255,255,000
Color for the column header displaying "descending" order.
255,000,000
Use application activation
Search
Save settings for future use
OK
Cancel


VFX - Application Builder - Vfx100traders
Show grid lines
2 - no in all forms
Grid Highlight Style
-1 – use grid settings
AutoFit grids on first load.
2 - Never use Auto Fit
Pressing the enter key on a grid switches the form into edit-mode
2 - False for all forms
Search dialog: use grid columns/use all fields
1 - use fields from grid in all forn
Indexes
Recreate temporary index files after editing
Display a wait window message while deleting temporary index files
Disable clearing indexes when editing data
Disable clearing indexes when inserting records
Disable clearing indexes when deleting records
Filtered index will be used instead of filtering
0 - Use form setting
Paths
Database folder
DATA
Use application activation
Search
Save settings for future use
OK
Cancel


VFX - Application Builder - Vfx100traders
Database name
TASTRADE.DBC
Metadata folder
data\Update\
Name of metadata table
Datadict
Default import folder
Current export folder
Path to the external report files (*.frx)
Save Export files folder per user
Misc
Name of Postscript printer to be installed when necessary
HP DeskJet 1200C/PS
Always install PS printer
Name of Fax printer driver to be used when sending fax reports
URL used when checking for internet connection existence
www.google.com
Use application activation
Search
Save settings for future use
OK
Cancel


VFX - Application Builder - Vfx100traders
Password to be used for encrypting config.vfx file
Support URL
Support e-mail
List separator chars
;
Security tables list
Install ClickYes
Author
Author:
Uwe Habermann
Company:
Address:
City:
Use application activation
Search
Save settings for future use
OK
Cancel


VFX - Application Builder - Vfx100traders
Install ClickYes
Author
Author:
Uwe Habermann
Company:
Address:
City:
State:
PostalCode:
Country
Company web site URL
Feedback email address
Use application activation
Search
Save settings for future use
OK
Cancel

Die Klasse *CApplication* ist die Klasse des Anwendungsobjekts. Die Eigenschaften und Methoden des Anwendungsobjekts stehen global in der gesamten Anwendung zur Verfügung.

Die Klasse *CApplication* wird in *Vfxmain.prg* programmatisch von der visuellen Klasse *CFoxappl* aus der Klassenbibliothek *Appl.vcx* abgeleitet. In dieser Klasse macht der VFX – Application Builder die Eintellungen. Hier können bei Bedarf auch Methoden überschrieben oder verändert werden. Die für die Steuerung der Anwendung wichtigen Eigenschaften des Anwendungsobjekts sollen hier im Einzelnen erläutert werden.

cAscOrderRGB – RGB-Wert einer Farbe, die verwendet wird um eine aufsteigende Sortierung in einer Grid-Spalte in der Überschrift anzuzeigen. Der Standardwert ist *"RGB(255,255,0)"*.

cDataDir - Der Pfad in dem sich die Datenbank befindet. Standardmäßig wird dieser Pfad aus der Konstanten *datapath_loc* aus der Datei *Userdef.h* gelesen. Weisen Sie dieser Eigenschaft einen Leerstring zu, wenn Sie Multi-Client-Database-Eigenschaft von VFX nutzen möchten.

cDateFormat – Das Datumsformat, das standardmäßig in der Anwendung verwendet wird. Der Wert dieser Eigenschaft wird als Parameter dem Befehl SET DATE übergeben. Der Wert dieser Eigenschaft wird normalerweise in der Methode *setlangid* des Anwendungsobjekts entsprechend der eingestellten Sprache zugewiesen.

cDescOrderRGB – RGB-Wert einer Farbe, die verwendet wird um eine absteigende Sortierung in einer Grid-Spalte in der Überschrift anzuzeigen. Der Standardwert ist *"RGB(255,0,0)"*.

cEdt_Date – Der Name eines Feldes in einer beliebigen Tabelle. Wenn ein Datensatz gespeichert wird und in der betreffenden Tabelle ein Feld mit diesem Namen gefunden wird, werden hier das Datum und ggf. die Uhrzeit der Bearbeitung gespeichert. Der Typ des Feldes kann *Date* oder *Datetime* sein. Der Standardwert ist ein Feld mit dem Namen *edt_date*.

cEdt_Usr – Der Name eines Feldes in einer beliebigen Tabelle. Wenn ein Datensatz gespeichert wird und in der betreffenden Tabelle ein Feld mit diesem Namen gefunden wird, wird hier der Name des Benutzers gespeichert, der den Datensatz verändert hat. Das Feld muss vom Typ Zeichen sein. Der Standardwert ist ein Feld mit dem Namen *edt_usr*.

cExcludeFiles – Hier kann eine durch Komma separierte Liste von Dateinamen eingegeben werden. Die hier aufgeführten Dateien erscheinen nicht im Dialog Datenbankwartung und sind von der Datenbank-wartung ausgeschlossen. Der Standardwert ist *"DBCXREG.DBF;CDBKMETA.DBF;SDTMETA.DBF;SDTUSER.DBF;COREMETA.DBF"*.

cHelpFile – Der Name der Hilfedatei, die beim drücken der Taste F1 geöffnet werden soll. Die Namenserweiterung (*chm* oder *hlp*) muss mit angegeben werden. Der Standardwert ist der Name des Projekts mit der Namenserweiterung *chm*.

cIns_Date – Der Name eines Feldes in einer beliebigen Tabelle. Wenn ein neuer Datensatz gespeichert wird und in der betreffenden Tabelle ein Feld mit diesem Namen gefunden wird, werden hier das Datum und ggf. die Uhrzeit der Neuanlage gespeichert. Der Typ des Feldes kann *Date* oder *Datetime* sein. Der Standardwert ist ein Feld mit dem Namen *ins_date*.

cIns_Usr – Der Name eines Feldes in einer beliebigen Tabelle. Wenn ein neuer Datensatz gespeichert wird und in der betreffenden Tabelle ein Feld mit diesem Namen gefunden wird, wird hier der Name des Benutzers gespeichert, der den Datensatz neu angelegt hat. Das Feld muss vom Typ *Zeichen* sein. Der Standardwert ist ein Feld mit dem Namen *ins_usr*.

cIntroBitmap – Der Name einer Bilddatei, die als Splashscreen angezeigt werden soll. Es sind alle von VFP unterstützten Grafikformate zulässig, also zum Beispiel *bmp*, *jpg*, *gif* oder *png*. Der Standardwert ist *Bitmap\Intro.png* und wird aus der Include-Datei *Userdef.h* gelesen.

cIntroForm – Der Name einer Formularklasse, die den Splashscreen anzeigen soll. Eine Änderung dieses Wertes ist nur erforderlich, wenn ein Splashscreen mit besonderen Eigenschaften verwendet werden soll. Der Standardwert ist *CSplashDialog*.

cLoginForm – Der Name einer Formulardatei, die den Anmeldedialog enthält. Eine Änderung dieser Eigenschaft ist nur erforderlich, wenn die Benutzerverwaltung von VFX nicht verwendet soll und ein eigenes Verfahren zur Benutzerverwaltung zum Einsatz kommt. Der Standardwert ist *Vfxlogin.scx*.

cMainDatabase – Der Name der Datenbank. Der Wert wird aus der Konstanten *database_loc* aus der Datei *Userdef.h* gelesen. Der Standardwert wurde mit dem VFX – Application Wizard beim Erstellen des Projekts festgelegt.

cMainForm – Der Name eines Formulars, das beim Start der Anwendung nach der Benutzeranmeldung angezeigt werden soll. Der Standardwert ist eine leere Zeichenkette.

cMainIcon – Das Symbol der Anwendung. Standardmäßig wird dieses Symbol in allen Formularen verwendet. Der Standardwert ist *Bitmap\Main.ico* und wird aus der Konstanten *mainicon_loc* aus der Include-Datei *Userdef.h* gelesen

cMainTitle – Der Name der Anwendung. Dieser Name erscheint in der Titelzeile der Anwendung. Der Name der Anwendung kann auch beim Befehl *CREATEOBJECT(„capplication“,<Name der Anwendung>)* als zweiter Parameter angegeben werden. In diesem Fall wird der Wert dieser Eigenschaft überschrieben.

cMainToolbar – Der Name der Standard-Symbolleiste. Der Standardwert wurde mit dem VFX – Application Wizard beim Anlegen des Projekts festgelegt. VFX stellt zwei Klassen mit Symbolleisten zur Verfügung. Die Klasse *CAppToolbar* enthält keine Schaltflächen zur Bewegung des Datensatzzeigers in Formularen. Die Klasse *CAppNavBar* enthält Schaltflächen zur Bewegung des Datensatzzeigers in Formularen.

cvfxpath – In dieser Eigenschaft kann der Name der Tabelle angegeben werden, die die Informationen zu den Pfaden der Datenbanken der Anwendung enthält. Der Standardwert ist *Vfxpath.dbf*.

FileMnuOffset – Dies ist die Nummer des Eintrags im Menü „*Datei*“, das für den ersten Eintrag eines zuletzt verwendeten Formulars verwendet wird. Wenn Sie dem Menü „*Datei*“ eigene Einträge hinzufügen wollen, muss dieser Wert entsprechend erhöht werden.

lAllowDeleteChildData - Wenn der Wert dieser Eigenschaft auf *wahr* gesetzt wird, dürfen Benutzer, die in OneToMany-Formulare keine Datensätze löschen dürfen, trotzdem Child-Datensätze löschen. Wenn dieser Wert auf *falsch* gesetzt wird, dürfen auch keine Child-Datensätze gelöscht werden.

lAutoLogin – Wenn der Wert dieser Eigenschaft auf *wahr* gesetzt wird, werden Benutzer, die in der Benutzerverwaltung registriert sind, beim Anwendungsstart ohne Aufforderung zur Eingabe eines Kennworts automatisch angemeldet. Der Standardwert dieser Eigenschaft ist *falsch*.

lCentury – Wenn der Wert dieser Eigenschaft auf wahr gesetzt ist, wird in allen Datumsfeldern der Anwendung die Jahreszahl 4stellig angezeigt. Der Standardwert ist *falsch*, Jahreszahlen werden 2stellig angezeigt.

lDisableFormResize – Wenn der Wert dieser Eigenschaft auf *wahr* gesetzt wird, ist das Ändern der Größe aller Formulare der Anwendung nicht möglich. Der Standardwert ist *falsch*, die Größe von Formularen kann vom Benutzer verändert werden.

lNoClearIdxOnDelete – Standardmäßig löscht VFX temporäre Indexdateien, wenn ein Datensatz gelöscht werden soll. Setzen Sie den Wert dieser Eigenschaft auf *wahr*, wenn temporäre Indexdateien in dieser Situation nicht gelöscht werden sollen. Beachten Sie, dass temporäre Indexdateien nicht geöffnet sein dürfen, wenn Transaktionen ausgeführt werden. Der Standardwert ist *falsch*.

lNoClearIdxOnEdit – Standardmäßig löscht VFX temporäre Indexdateien, wenn ein Datensatz bearbeitet werden soll. Setzen Sie den Wert dieser Eigenschaft auf *wahr*, wenn temporäre Indexdateien in dieser

Situation nicht gelöscht werden sollen. Beachten Sie, dass temporäre Indexdateien nicht geöffnet sein dürfen, wenn Transaktionen ausgeführt werden. Der Standardwert ist *falsch*.

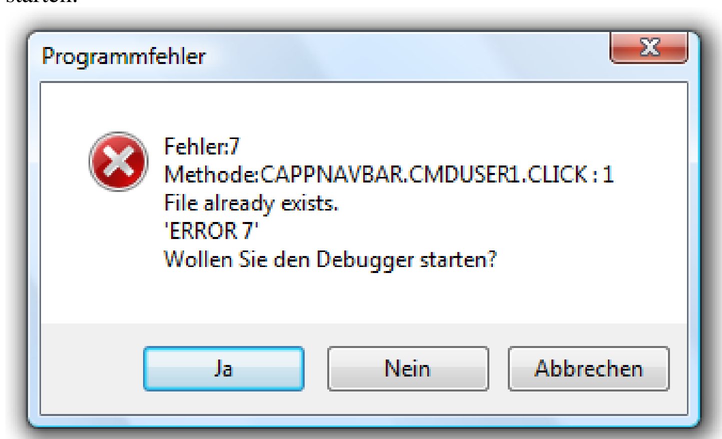
lNoClearIdxOnInsert – Standardmäßig löscht VFX temporäre Indexdateien, wenn ein Datensatz neu angelegt werden soll. Setzen Sie den Wert dieser Eigenschaft auf *wahr*, wenn temporäre Indexdateien in dieser Situation nicht gelöscht werden sollen. Beachten Sie, dass temporäre Indexdateien nicht geöffnet sein dürfen, wenn Transaktionen ausgeführt werden. Der Standardwert ist *falsch*.

lRelogonQuit – Steuert das Verhalten der Anwendung, wenn ein Benutzer versucht sich während die Anwendung läuft erneut anzumelden und den Vorgang abbricht. Wenn der Wert dieser Eigenschaft auf *wahr* gesetzt wird, wird die Anwendung beendet. Wenn der Wert dieser Eigenschaft auf *falsch* gesetzt wird, bleibt der zuletzt angemeldete Benutzer angemeldet.

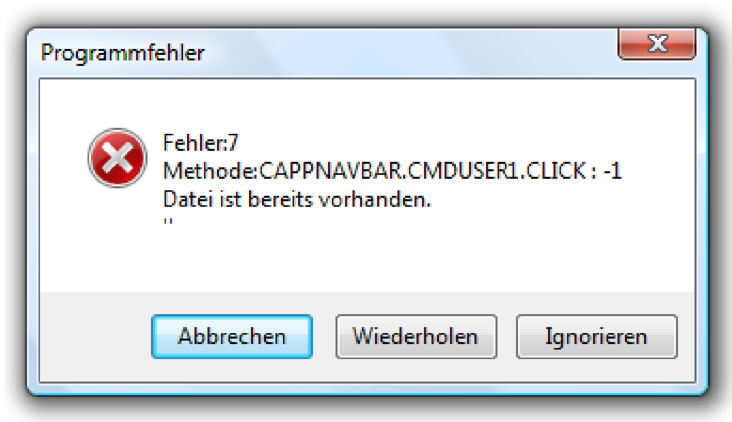
lRemakeIdxAfterClear – Wenn der Wert dieser Eigenschaft auf *wahr* gesetzt wird, werden temporäre Index-dateien nach dem Abschluss eines Speichervorgangs automatisch wieder angelegt. Vergleichen Sie auch mit den Eigenschaften *lNoClearIdxOnDelete*, *lNoClearIdxOnEdit*, *lNoClearIdxOnInsert*. Der Standardwert dieser Eigenschaft ist *falsch*.

nAppOnErrorBehavior – Diese Eigenschaft steuert das Verhalten der Anwendung im Fehlerfall.
0 – Laufzeitfehler werden ignoriert.
1 – Es wird eine Fehlermeldung angezeigt (Standardwert).

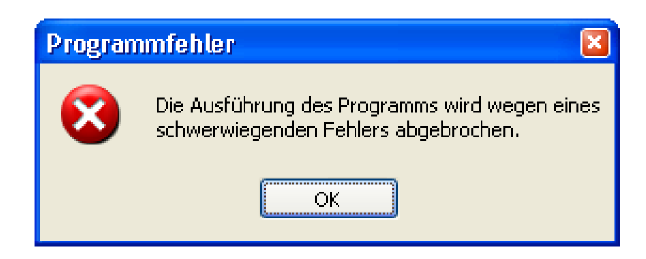
Bei einem Laufzeitfehler in der Entwicklungsumgebung hat der Entwickler die Möglichkeit den Debugger zu starten:



Bei einem Laufzeitfehler in der ausfürbaren Datei wird ein Laufzeitfehler so angezeigt:



2 – Die Ausführung der Anwendung wird nach Anzeige eines Hinweises beendet.



ErrorDetailLevel – Diese Eigenschaft steuert welche Informationen im Fehlerfall in der Tabelle *Vfxlog.dbf* protokolliert werden.
0 – Nur die Fehlermeldung aber keine Information über den Aufrufstapel.
1 – Die Fehlermeldung und Informationen über den Aufrufstapel (Standardwert).
2 – Vollständige, detaillierte Fehlerinformationen.

PSPrinterToInstall – Diese Eigenschaft enthält den Namen des Standard-Postscript-Druckertreibers. Dieser Druckertreiber wird automatisch installiert, wenn noch kein Postscript-Druckertreiber installiert ist und die Anwendung einen Postscript-Druckertreiber braucht um eine PDF-Datei zu erstellen. Der Standardwert ist "HP DeskJet 1200C/PS".

cConnectionCheckURL – Diese Eigenschaft enthält die Adresse einer Internetseite, die verwendet wird um zu testen, ob eine Internet-Verbindung besteht. Diese Eigenschaft wird benötigt wenn Ghostscript nicht installiert ist. Ghostscript wird bei Bedarf automatisch aus dem Internet heruntergeladen und installiert. Ghostscript wird verwendet um Postscript-Dateien in PDF-Dateien umzuwandeln. Wenn keine Verbindung mit dem Internet besteht und auch keine DFÜ-Netzwerkverbindung eingerichtet ist, wird von VFX ein Eintrag im DFÜ-Netzwerk angelegt. Alle Eigenschaften der DFÜ-Verbindung können vom Entwickler vorgegeben werden. Der Anwender kann bei Bedarf in einem Dialog die Telefonnummer, den Benutzernamen und das Kennwort ändern.

lUseActivation – Über diese Eigenschaft wird die Produktaktivierung ein- bzw. ausgeschaltet. Diese Eigenschaft kann im VFX – Application Wizard eingestellt werden, wenn ein neues Projekt erstellt wird. Später kann der Eintrag im VFX – Application Builder geändert werden. Der Standardwert ist .F., die Produktaktivierung wird nicht verwendet.

lActivationType – Wenn diese Eigenschaft auf .T. gesetzt wird, überprüft die Klasse *CVFXActivate* ob die Datei *FirstInstall.txt* existiert, wenn die Anwendung gestartet wird. Diese Eigenschaft kann im VFX Application Wizard eingestellt werden, wenn ein neues Projekt erstellt wird. Der Standardwert ist .F., es wird nicht auf das Vorhandensein der Datei *FirstInstall.txt* geprüft.

cConfigPassword – Kennwort für die Verschlüsselung der Datei *Config.vfx*. Dieses Kennwort wird aus Sicherheitsgründen benötigt. Die Verbindungsinformationen zur Datenquelle, wie Benutzername und Kennwort, sind so auch für versierte Anwender nicht im Klartext einsehbar.

cFaxPrinterName – Der Name des Fax-Druckertreibers, der zur Versendung von Berichten als Fax verwendet werden soll. Wenn dieser Wert leer ist, versucht VFX einen Druckertreiber mit „Fax“ im Namen zu finden. Bevorzugt werden die Druckertreiber Winfax und Fritz!fax verwendet.

cMetadataTableName – Name der Tabelle mit den Metadaten. Diese Tabelle wird zur Aktualisierung einer SQL Server-Datenbank beim Kunden benötigt. Der Standardwert ist *Datadict*.

lAllowMultipleLogin – Wenn diese Eigenschaft auf .T. eingestellt ist, dürfen sich Benutzer mehrmals gleichzeitig an der Anwendung anmelden. Der Standardwert ist .T.

lAllowUserCustomization – Wenn diese Eigenschaft auf .T. eingestellt ist, können die Umgebungseinstellungen je Benutzer gespeichert werden. Der Standardwert ist .T.

lInformUserForUpdate – Wenn diese Eigenschaft auf .T. eingestellt ist, wird vor der Aktualisierung der Kundendatenbank eine Meldung angezeigt. Der Standardwert ist .F.

lSaveExportPathPerUser – Wenn diese Eigenschaft auf .T. eingestellt ist, wird der Exportpfad für PDF-, BMP-, HTML- und TIFF-Dateien je Benutzer in der Ressourcentabelle *Vfxres.dbf* gespeichert. Der Standardwert ist .T.

lShowProgressOnUpdate – Wenn diese Eigenschaft auf .T. eingestellt ist, wird während der Aktualisierung der Kundendatenbank eine Fortschrittsanzeige angezeigt. Der Standardwert ist .T.

lUseBCCRecipients – Wenn diese Eigenschaft auf .T. eingestellt ist, wird im Dialog zur Eingabe von E-Mail-adressen eine Textbox zur Eingabe von BCC-Empfängern angezeigt. Der Standardwert ist .T. Dieses Feature wird nicht von allen E-Mailprogrammen unterstützt! BCC funktioniert zum Beispiel mit Outlook Express, nicht jedoch mit Outlook.

nDockable – Einstellung des Dock-Verhaltens von Formularen. -1 – Die Einstellung des Formulars wird verwendet, 0 – Alle Formulare können nicht gedockt werden, 1 – Alle Formulare unterstützen „Docking“, 2 – Alle Formulare unterstützen „Docking“ sind aber nicht dockbar. In diesem Fall können Formulare nur ineinander gedockt werden. Modale Formulare können grundsätzlich nicht gedockt werden.

nHighLightStyle – Mit dieser Eigenschaft kann die Eigenschaft *HighlightStyle* von der Klasse *CGrid* global eingestellt werden.

nIndexInsteadOfFilter – Mit dieser Eigenschaft kann eingestellt werden, ob anstelle von Filtern mit gefilterten, temporären Indexdateien gearbeitet werden soll. 0 - Die Einstellung des Formulars wird verwendet, 1- immer gefilterten, temporären Indexdateien verwenden, 2 – es wird immer mit Filtern gearbeitet.

nNullValid – Mit dieser Eigenschaft kann eingestellt werden, ob Eingaben in Auswahlfeldern erforderlich sind. 0 - Die Einstellung des Auswahlfeldes wird verwendet, 1 – eine leere Eingabe ist erlaubt, 2 - eine leere Eingabe ist nicht erlaubt.

nSearchOnInit – Mit dieser Eigenschaft kann eingestellt werden, ob beim Start eines Formulars der zuletzt verwendete Filter gesetzt werden soll.

8.2. VFX – Form Wizard

Wie in bisherigen VFX-Versionen sollte der VFX – Form Wizard zum Erstellen neuer Formulare verwendet werden. Die Bedienung des VFX – Form Wizard wurde bereits im Kapitel „Schnelleinstieg” erläutert. Als Erweiterung zum Verhalten des Form Builders in VFX 8.0 wird jetzt automatisch nach der Erstellung eines Formulars im VFP Formular-Designer der VFX – Form Builder gestartet. Die VFX Formular Builder beinhalten den neuen VFX – Data Environment Builder. Der Entwickler wird also Schritt für Schritt von der Auswahl einer geeigneten Formularklasse bis zum lauffähigen Formular Builder-unterstützt geführt.

8.3. VFX – Form Builder

Die VFX – Form Builder unterstützen alle neuen Formulareigenschaften von VFX 11.0. Die Formular Builder in VFX 11.0 wurden grundlegend überarbeitet und um zahlreiche Funktionen erweitert. Zusätzlich können jetzt viele Funktionen über die Form Builder eingestellt werden, die bisher nur manuell in VFP bearbeitbar waren. Auf neuen Seiten der Form Builder können Ansichtsparameter, in Beziehung stehende Tabellen, erforderlich Eingabefelder und Felder für Berichte bearbeitet werden.

Die größte Neuerung ist der im ersten Dialogschritt des Formular Builders erscheinende Data Environment Builder.

8.4. VFX – Dataenvironment Builder

Die VFX – Form Builder ermöglichen dem Entwickler neben dem Layout der Formulare auch die Datenumgebung zu bearbeiten.

Der Datenumgebung können Tabellen, Ansichten oder bestehende CursorAdapter-Klassen hinzugefügt werden oder auch neue CursorAdapter-Klassen erstellt werden. Es können Indexschlüssel für CursorAdapter erstellt werden und es können Beziehungen zwischen Cursor-Objekten eingerichtet werden.

Auf der Seite *Aliases* können Cursor-Objekte hinzugefügt oder erstellt werden.

Mit einem Klick auf die Schaltfläche *Add* können bestehende Tabellen oder Ansichten der Datenumgebung hinzugefügt werden. Der VFP-Dialog zur Auswahl von Tabellen und Ansichten wird geöffnet. Wenn ein Cursor in der Datenumgebung auf einer Tabelle basiert, kann in der Spalte *Order* ein Index der Tabelle gewählt werden.

Über die Schaltfläche *Add CA* kann ein CursorAdapter, basierend auf einer CursorAdapter-Klasse, hinzugefügt werden. Eine solche CursorAdapter-Klasse kann zum Beispiel mit dem VFX – CursorAdapter Wizard erstellt werden.

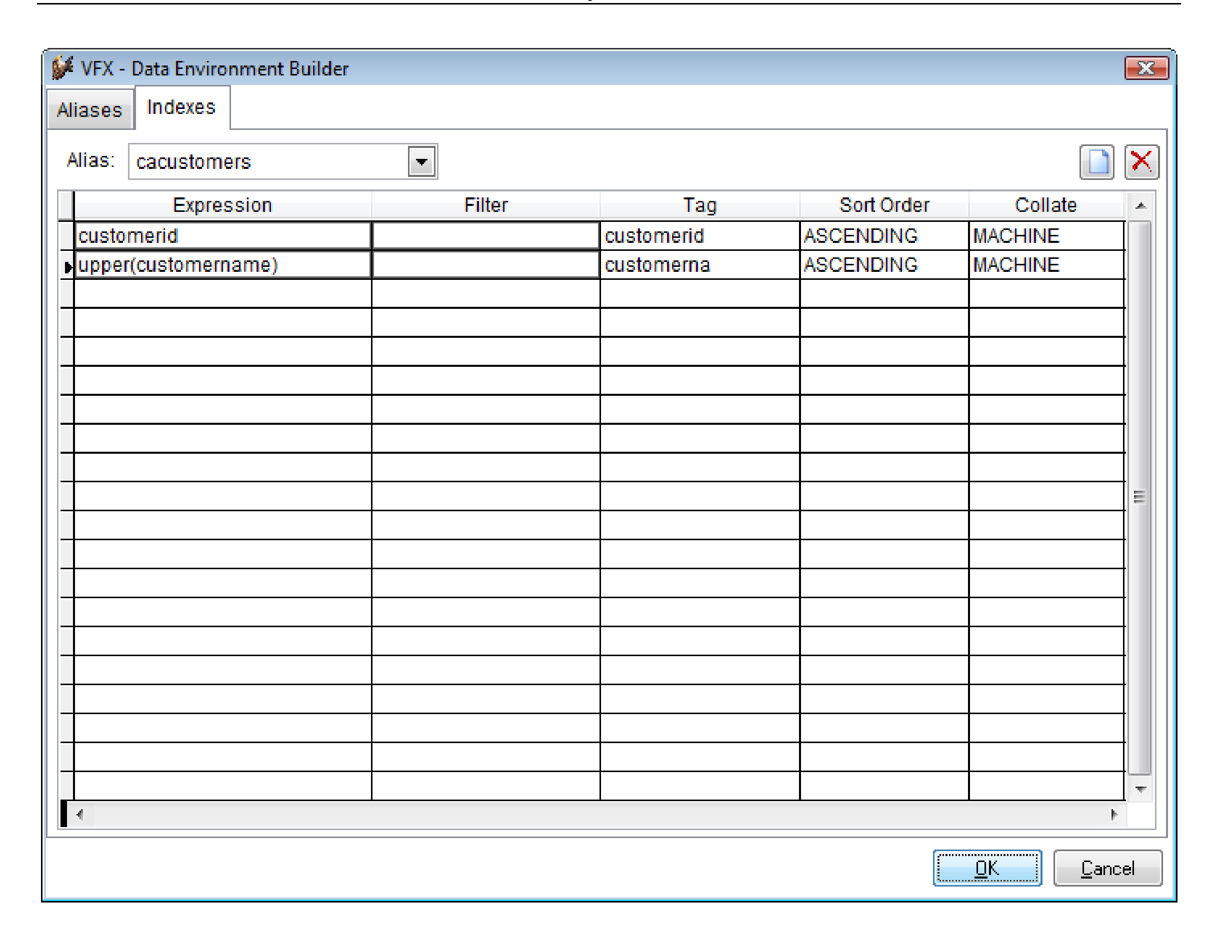
Über die Schaltfläche *New CA* kann ein neues Objekt basierend auf der Klasse *CAppDataAccess* mithilfe des VFP – CursorAdapter Builder erstellt werden.

Wenn der Cursor in der Datenumgebung auf einer Tabelle basiert, kann in der Spalte *Order* eine Sortierfolge aus den existierenden Indexschlüsseln ausgewählt werden. Wenn der Cursor auf einem CursorAdapter basiert, kann ein Indexausdruck aus einer Liste der für diesen CursorAdapter definierten Indexausdrücke ausgewählt werden. Die Indexschlüssel werden zur Laufzeit erstellt, nachdem der CursorAdapter mit Daten gefüllt wurde. Indexschlüssel für CursorAdapter können auf der Seite *Indexes* angelegt werden.

Die Namen und Aliasnamen der Cursor in der Datenumgebung können beliebig geändert werden.

In der Spalte *Filter* kann ein logischer Ausdruck eingegeben werden, der zur Laufzeit als Filterausdruck verwendet wird. Dieser Ausdruck wird der Eigenschaft *Filter* des Cursor-Objekts zugewiesen.

Die Spalten *Parent Alias* und *Rel Expression* geben die Möglichkeit Relationen zwischen Cursorn in der Datenumgebung aufzubauen. Nach Auswahl eines Aliasnamen in der Spalte *Parent Alias*, kann in der Spalte *Rel.Expression* ein Feld für den Relationsausdruck ausgewählt werden oder es kann ein eigener Relationsausdruck eingegeben werden. Zur Laufzeit werden die Beziehungen vom *oRelationMgr*-Objekt verwaltet.



Um zwischen CursorAdapter-Objekten Beziehungen herstellen zu können, müssen temporäre Indexschlüssel zur Laufzeit erstellt werden. Auf der Seite *Indexes* kann der Entwickler die erforderlichen Indexschlüssel erstellen. VFX erstellt die entsprechenden Indexdateien temporär zur Laufzeit und erstellt die Beziehungen, die auf der Seite *Aliases* eingegeben wurden.

Für Cursor-Objekte, die auf Tabellen basieren, werden die zur Verfügung stehenden Indexschlüssel angezeigt. Für CursorAdapter-Objekte können die Indexschlüssel bearbeitet und neue Indexschlüssel hinzugefügt werden.

Für jeden zu erstellenden Indexschlüssel müssen der Tag-Name, der Indexausdruck und die Sortierfolge eingegeben werden. Wenn ein gefilterter Indexschlüssel gewünscht wird, kann der Filterausdruck in der Spalte Filter eingegeben werden.

Durch einen Klick auf die Schaltfläche „Next“ gelangt man zum VFX – Form Builder.

8.5. VFX – CDataFormPage Builder

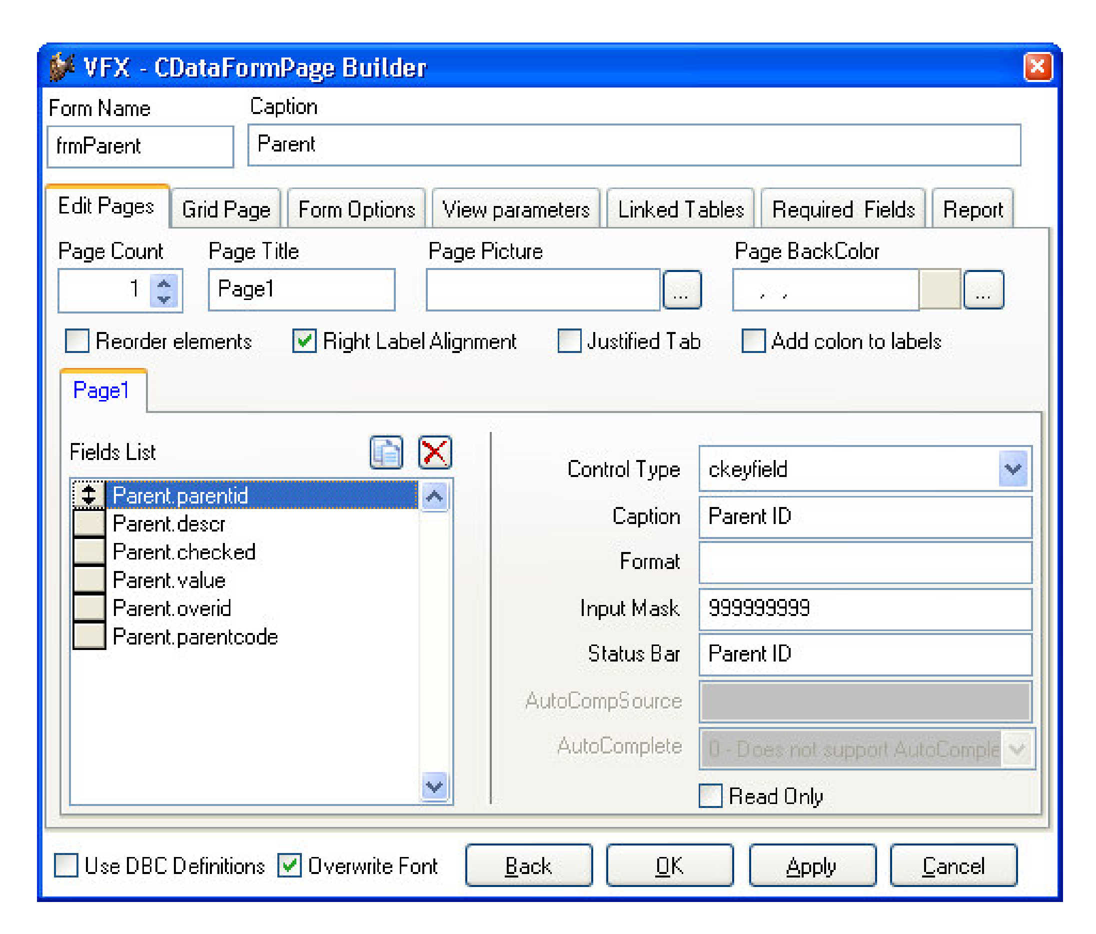
Um einen VFX-Formular-Builder aufzurufen, bewegen Sie die Maus auf den weißen Hintergrund des Formular-Designers, drücken Sie die rechte Maustaste und wählen Sie Builder.

Der VFX – CDataFormPage Builder wird geladen und zeigt einen benutzerfreundlichen Dialog:

8.5.1. Edit Pages

Im VFX Form Builder können auf der Seite *Edit Pages* alle neuen Formulareigenschaften von VFX 11.0 wie Hintergrundbild oder Hintergrundfarbe für Seiten eines Seitenrahmens, verknüpfte Tabellen und erforderliche Felder sowie AutoComplete-Eigenschaften eingestellt werden.

Wenn das Kontrollkästchen *Add colon to labels* markiert wird, wird an alle Labels ein Doppelpunkt angefügt.



Form Name. Geben Sie den Namen des neuen Formulars ein. Der VFX – Form Wizard hat bereits einen Standardnamen entsprechend den Namenskonventionen zugewiesen. Der Name beginnt mit *frm*. Selbstverständlich können Sie Ihrem Formular einen beliebigen Namen geben, aber wir empfehlen Ihnen, sich an die allgemeinen Namenskonventionen zu halten.

Caption. Geben Sie die Überschrift für Ihr Formular ein. Während Sie die Überschrift eingeben, wird diese bereits in der Überschrift des Formular-Builders angezeigt. Wenn Ihr Formular veränderliche Überschriften in Abhängigkeit vom Aufruf des Formulars haben soll, brauchen Sie sich um diese Überschrift keine Gedanken zu machen. Geben Sie in diesem Fall einfach eine mehr oder weniger zutreffende Überschrift ein.

Page Count. Geben Sie ein, wie viele Bearbeitungsseiten Sie benötigen. Für einige Formulare wird eine Bearbeitungsseite ausreichend sein. Wenn Sie mehr Felder haben, werden Sie diese auf mehrere Seiten verteilen wollen. In Abhängigkeit von der Anzahl der gewählten Seiten, sehen Sie im Seitenrahmen des Formular-Builder einen Seitenrahmen, der diese Seiten anzeigt. Wenn Sie zwei Bearbeitungsseiten eingeben, sehen Sie zwei Seiten auf dem Seitenrahmen, wenn Sie drei Bearbeitungsseiten eingeben, sehen Sie drei Seiten auf dem Seitenrahmen usw.

Page Title. Geben Sie die Überschrift der aktuellen Bearbeitungsseite ein. Wenn Sie die Überschrift für die zweite Seite eingeben wollen, drücken Sie auf die zweite Seite und Sie können die Überschrift auch für diese Seite eingeben. Der VFX-Formular-Builder zeigt während der Eingabe die sich ergebende Überschrift für die einzelnen Seiten an.

Justified Tab. Markieren Sie dieses Kontrollkästchen, wenn die Seitenüberschriften justiert sein sollen. Ansonsten haben die Überschriften eine variable Länge und füllen nicht die Breite des Seitenrahmens.

Für jede Bearbeitungsseite stehen die folgenden Optionen zur Verfügung:

Fields Selected. Hier sehen Sie alle Felder, die Sie für die aktuelle Bearbeitungsseite ausgewählt haben. Um Felder hinzuzufügen benutzen Sie das *Field Assistant*-Fenster, das in einem eigenen Formular angezeigt wird und alle aus der Datenumgebung zur Verfügung stehenden Felder anzeigt.

Control Type. Geben Sie für alle ausgewählten Felder den zu benutzenden Steuerungstyp an. Zur Auswahl stehen alle von VFX angebotenen Klassen für Steuerelemente zur Verfügung.

ANMERKUNG: Um Ihre eigenen Klassen zu verwenden, tragen Sie diese im Datenbank-Container bei jedem Feld bei „Bibliothek für Anzeige" ein!

Caption. Überschrift für das ausgewählte Feld. Der Standardwert wird aus dem Datenbank-Container übernommen.

Format. Format-Eigenschaft für das selektierte Feld. Der Standardwert wird aus dem Datenbank-Container übernommen.

Input Mask. Eingabemasken-Eigenschaft für das selektierte Feld. Der Standardwert wird aus dem Datenbank-Container übernommen.

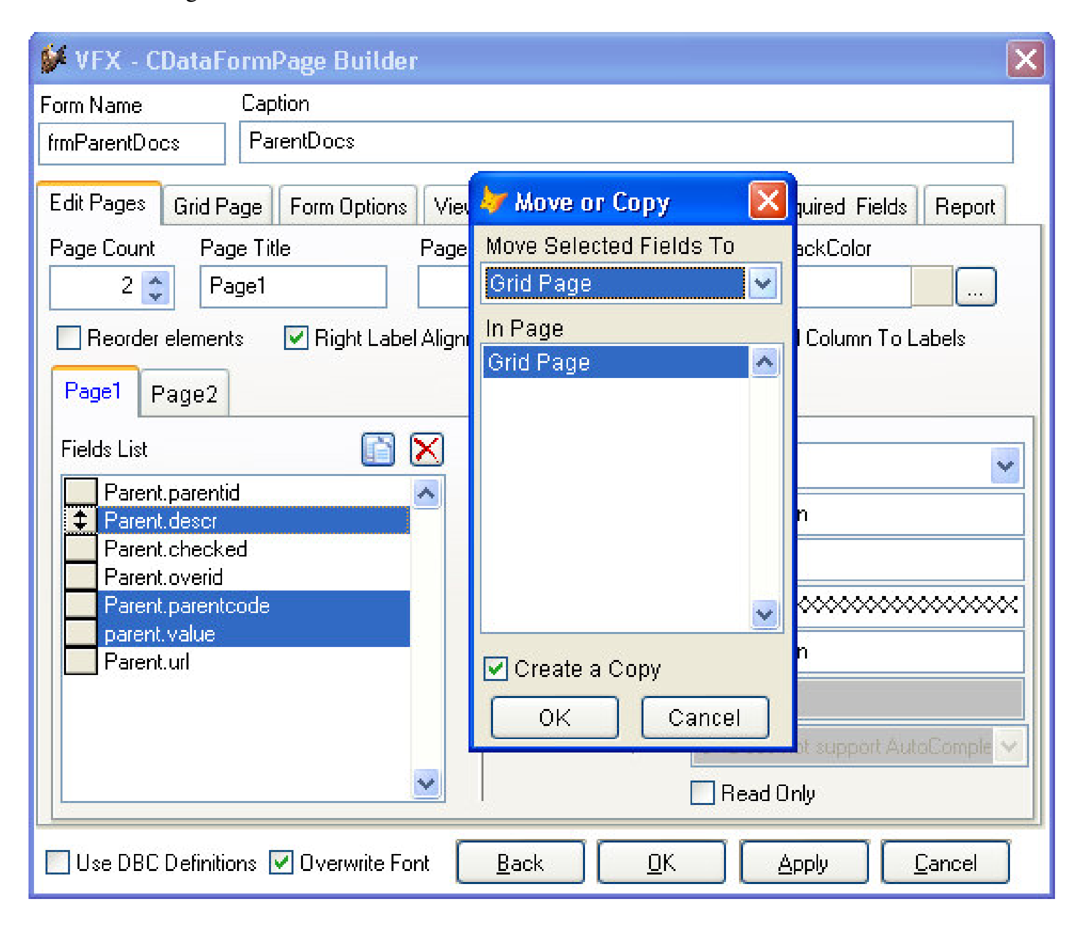
Status Bar. Meldung für die Statuszeile für dieses Feld. Der Standardwert wird aus dem Datenbank-Container übernommen. (Eigenschaft Feldkommentar, wenn dieser Wert leer ist, wird die Feldüberschrift genommen).

AutoCompSource. Name der Tabelle, die für die AutoComplete-Funktion in diesem Feld verwendet werden soll. AutoComplete-Tabellen müssen mit der Anwendung nicht ausgeliefert werden. Diese Tabellen werden von VFP bei Bedarf automatisch erstellt.

AutoComplete. Wert der Eigenschaft AutoComplete. Die AutoComplete-Funktion steht nur bei Textboxen zur Verfügung.

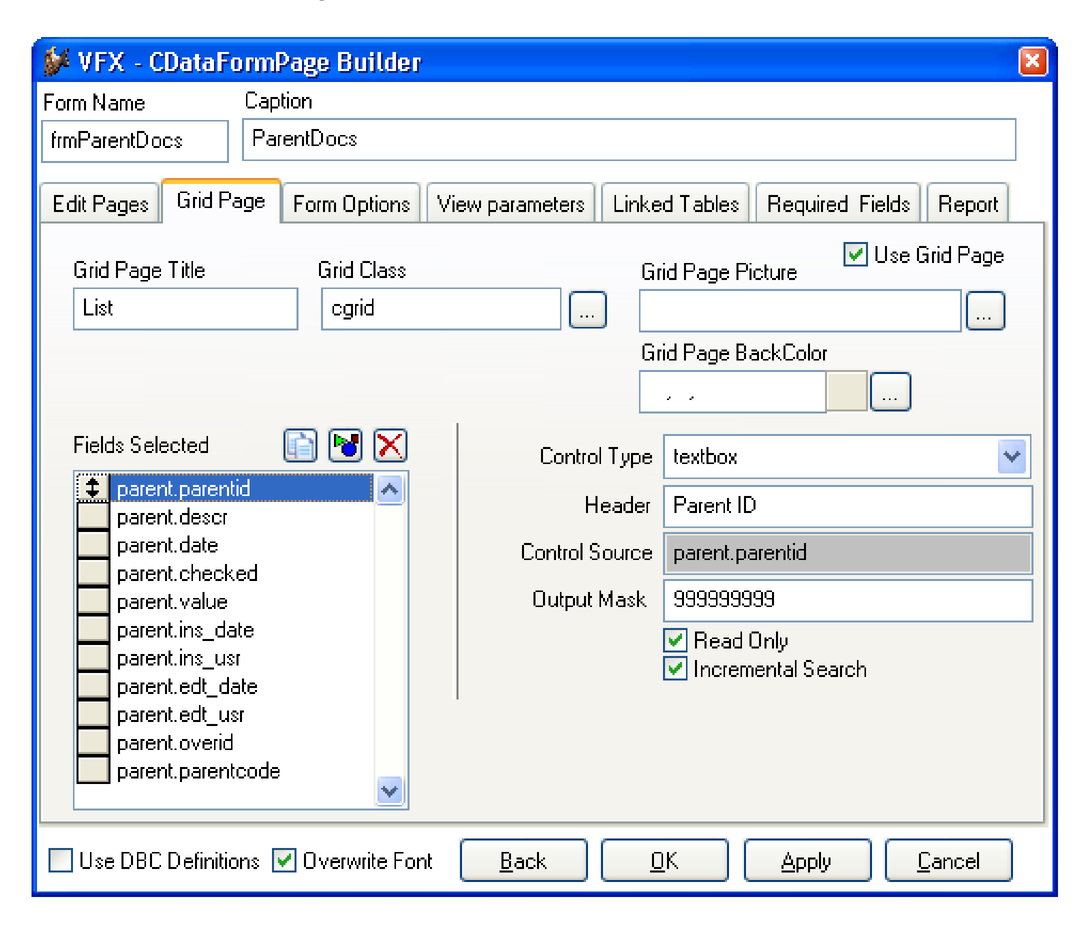
Read only. Wenn ein Steuerelement nur zur Anzeige von Informationen verwendet wird, markieren Sie dieses Kontrollkästchen.

Bei der Bearbeitung vorhandener Formulare ist die neue Schaltfläche *Move or Copy Fields* sehr nützlich. In der Feldliste können beliebig viele Felder markiert werden. Mithilfe des Dialogs *Move or Copy* können die markierten Felder auf eine andere Seite des Seitenrahmens kopiert oder verschoben werden. Die Zielseite kann eine Bearbeitungsseite, die Listenseite oder die Berichtsseite sein.



Wenn die ausgewählten Steuerelemente kopiert und nicht verschoben werden sollen, wird eine Kopie der Steuerelemente mit allen Eigenschaften auf der gewählten Seite angelegt.

8.5.2. Grid Page



Die folgenden Optionen stehen auf der Seite *Grid Page* zur Verfügung:

Use Grid Page. Markieren Sie dieses Kontrollkästchen, wenn Sie eine Listenseite auf Ihrem Formular haben wollen.

Grid Page Title. Geben Sie die Überschrift für die letzte Seite Ihres Formulars ein, die normalerweise ein Grid mit allen Datensätzen Ihrer Tabelle oder Ansicht enthält.

Grid Class. Geben Sie die Klasse für das Grid ein oder benutzen Sie den Standardwert, die *CGrid*–Klasse.

Fields Selected. Hier sehen Sie alle für das Grid ausgewählten Felder. Um Felder auszuwählen, benutzen Sie das *Field Assistant*-Fenster, in dem alle Felder aus der Datenumgebung zur Auswahl stehen.

Calculated Fields. Drücken Sie auf diese Schaltfläche um ein beliebiges berechnetes Feld hinzuzufügen.

Control Type. Geben Sie für alle ausgewählten Felder den gewünschten Kontrolltyp an.

Header. Überschriften für die Spalten Ihres Grids. Die VFX Formular-Builder fügen automatisch die Überschriften aus dem Datenbank-Container ein.

Output Mask. Die VFX-Formular-Builder erstellen die Ausgabemaske anhand der Feldlänge. Sie können die Ausgabemaske ändern, um sie an Ihre Bedürfnisse anzupassen.

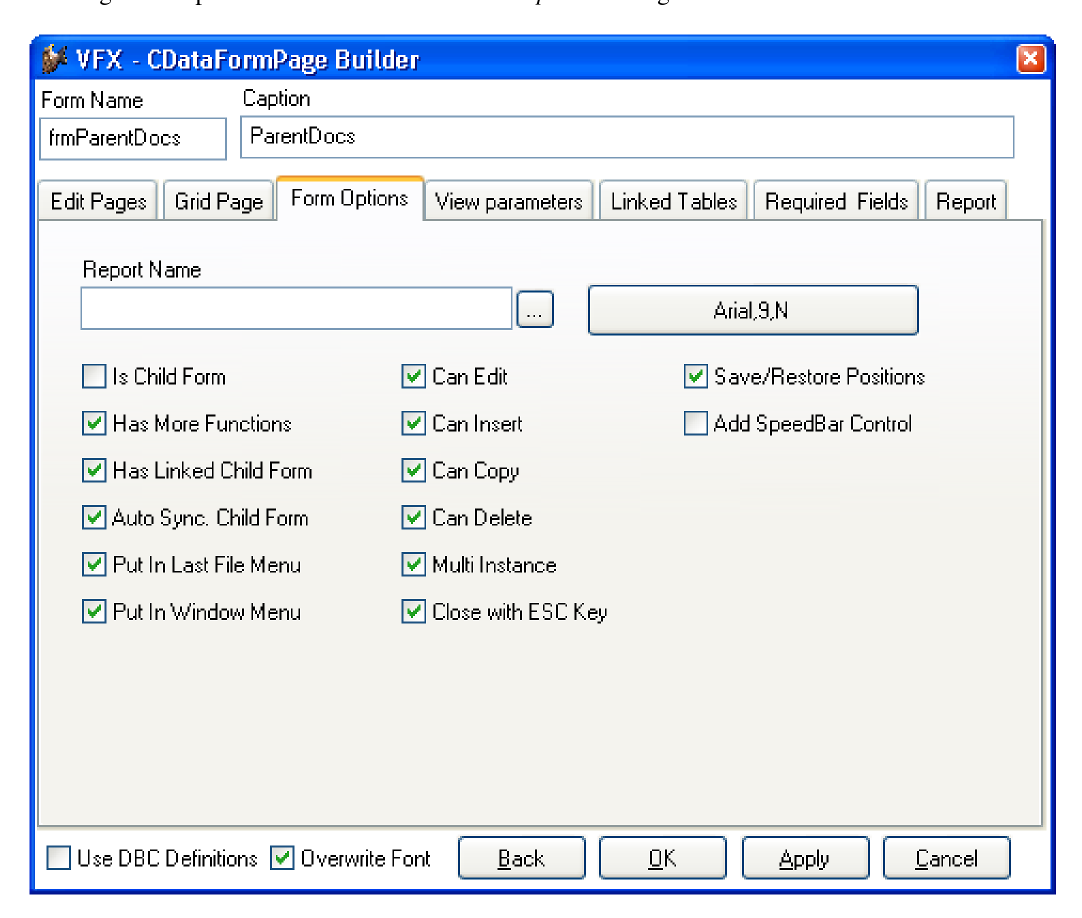
Read only. Wenn ein Steuerelement nur zur Anzeige von Informationen verwendet wird, markieren Sie dieses Kontrollkästchen.

Incremental Search. Markieren Sie dieses Kontrollkästchen, wenn Sie die inkrementelle Suche für die ausgewählte Spalte aktivieren wollen. Beachten Sie, dass VFX eine temporäre Indexdatei erstellt, wenn kein Indexschlüssel für die Spalte vorhanden ist. (Mit der *CGrid*-Eigenschaft *nMaxRec* können Sie angeben ab welcher Anzahl Datensätze dem Benutzer eine Meldung angezeigt werden soll, bevor eine temporäre Indexdatei erstellt wird.)

Zusätzlich gibt es auf dem VFX – Form Builder vier neue Seiten um die neuen Eigenschaften der VFX Formularklassen bearbeiten zu können.

8.5.3. Form Options

Die folgenden Optionen sind auf der Seite *Form Options* verfügbar:



Report Name. Hier können Sie den Namen eines Berichts eingeben. Wenn der Benutzer *drucken* oder *Seitenansicht* wählt, wird dieser Bericht gedruckt bzw. angezeigt. Sie brauchen für diese Funktionalität keinen Code in die Methode *OnPrint()* einzufügen. Wenn diese Eigenschaft leer gelassen wird, sucht VFX nach einem Bericht, der den gleichen Namen wie das Formular hat.

Is Child Form. Wenn das Formular, das Sie gerade erstellen, von einem anderen Formular aufgerufen wird, ist dieses Formular ein Child-Formular.

ANMERKUNG: Bitte verwechseln Sie dies nicht mit dem später beschriebenen 1:n-Formular, wo Sie die Haupttabelle und die Child-Tabelle **auf dem gleichen Formular** bearbeiten können. Hier sprechen wir über folgendes Verhalten: Formular 1 ruft Formular 2 auf, wobei Formular 1 das Hauptformular und Formular 2 das Child-Formular ist. Im Formular 2 sehen Sie nur die Datensätze, die ein bestimmtes Kriterium erfüllen, das die Verbindung zur Haupttabelle im Formular 1 herstellt.

Wenn Sie beispielsweise in einem Formular die Aufträge eines Kunden anzeigen wollen, markieren Sie dieses Kontrollkästchen und der VFX-Formular-Builder wird das Formular automatisch als Child-Formular erstellen. Dabei werden automatisch die erforderlichen Codezeilen in das *Init()*-Ereignis des Formulars eingetragen.

Für weitere Details lesen Sie bitte im Abschnitt *VFX – Parent/Child Builder* weiter unten in diesem Handbuch nach.

ANMERKUNG: Wenn Sie ein Formular haben, das sowohl als Child-Formular als auch als normales Formular dienen soll, markieren Sie die Option *Is Child Form*. Sie brauchen hierfür nicht zwei Formulare zu erstellen. Ein Formular kann sowohl alle Aufträge darstellen als auch nur die Aufträge eines bestimmten Kunden.

Has More Functions. Wenn das Formular, das Sie gerade erstellen, andere Formulare aufrufen oder Aktionen ausführen soll, müssen Sie dieses Kontrollkästchen markieren. Dadurch wird automatisch der erforderliche Code für die *OnMore()*-Methode Ihres Formulars erstellt. Sie müssen nur noch den Code in der *OnMore()*-Methode an Ihre Bedürfnisse anpassen. Normalerweise werden Sie eine Anzahl von Aktionen haben, die zur Auswahl in einem Formular angeboten werden. Der Benutzer kann dann die gewünschte Aktion auswählen.

Has Linked Child Form. Wenn das Formular, das Sie gerade erstellen, Child-Formulare aufrufen soll, die dynamisch mit diesem Hauptformular verbunden bleiben, markieren Sie dieses Kontrollkästchen. Dadurch wird automatisch der Code für die Formularmethode *OnSetChilddata()* erstellt. Diese Methode wird automatisch für jedes vorhandene Child-Formular aufgerufen.

Autosynch Child Form. Hiermit wird die Formulareigenschaft *lAutosynchChildform* festgelegt. Dadurch wird angegeben, ob die Child-Formulare automatisch mit diesem Hauptformular synchronisiert werden, wenn Sie den Datensatzzeiger im Hauptformular bewegen.

Put in Last File Menu. Hiermit wird die Formulareigenschaft *lPutinLastFile* festgelegt. Sie gibt an, ob die Formularüberschrift in die Liste der benutzen Dateien im Menü *Datei* eingetragen werden soll.

Put in Window Menu. Hiermit wird die Formulareigenschaft *lPutinWindowmenu* festgelegt. Sie gibt an ob das laufende Formular in das Menü *Fenster* eingetragen werden soll. Beachten Sie auch die Eigenschaft *nWinMnuCount* und die Methode *RefreshWindowMenu()* im Anwendungsobjekt.

Can Edit. Hiermit wird die Formulareigenschaft *lCanEdit* festgelegt. Sie gibt an, ob der Benutzer Datensätze im aktuellen Formular bearbeiten kann.

Can Insert. Hiermit wird die Formulareigenschaft *lCanInsert* festgelegt. Sie gibt an, ob der Benutzer Datensätze im aktuellen Formular einfügen kann.

Can Copy. Hiermit wird die Formulareigenschaft *lCanCopy* festgelegt. Sie gibt an, ob der Benutzer Datensätze im aktuellen Formular kopieren kann.

Can Delete. Hiermit wird die Formulareigenschaft *lCanDelete* festgelegt. Sie gibt an, ob der Benutzer Datensätze im aktuellen Formular löschen kann.

Multi Instance. Hiermit wird die Formulareigenschaft *lMultiInstance* eingestellt. Standardmäßig können alle Formulare, die Sie mit VFX erstellen, mehrmals geöffnet werden (das nennt man multiinstanzfähig). Dies ist eine großartige Eigenschaft. Alles was Sie dabei beachten müssen ist, dass das Formular mit einer privaten Datensitzung arbeiten muss. Dies ist der Standardwert in allen VFX-Formularen.

Trotzdem ist es manchmal günstig, die Eigenschaft multiinstanzfähig ausschalten zu können. Daher haben wir die Eigenschaft *lMultiInstance* eingeführt. Setzen Sie diese Eigenschaft auf *.F.* und das Formular kann nur einmal geöffnet werden.

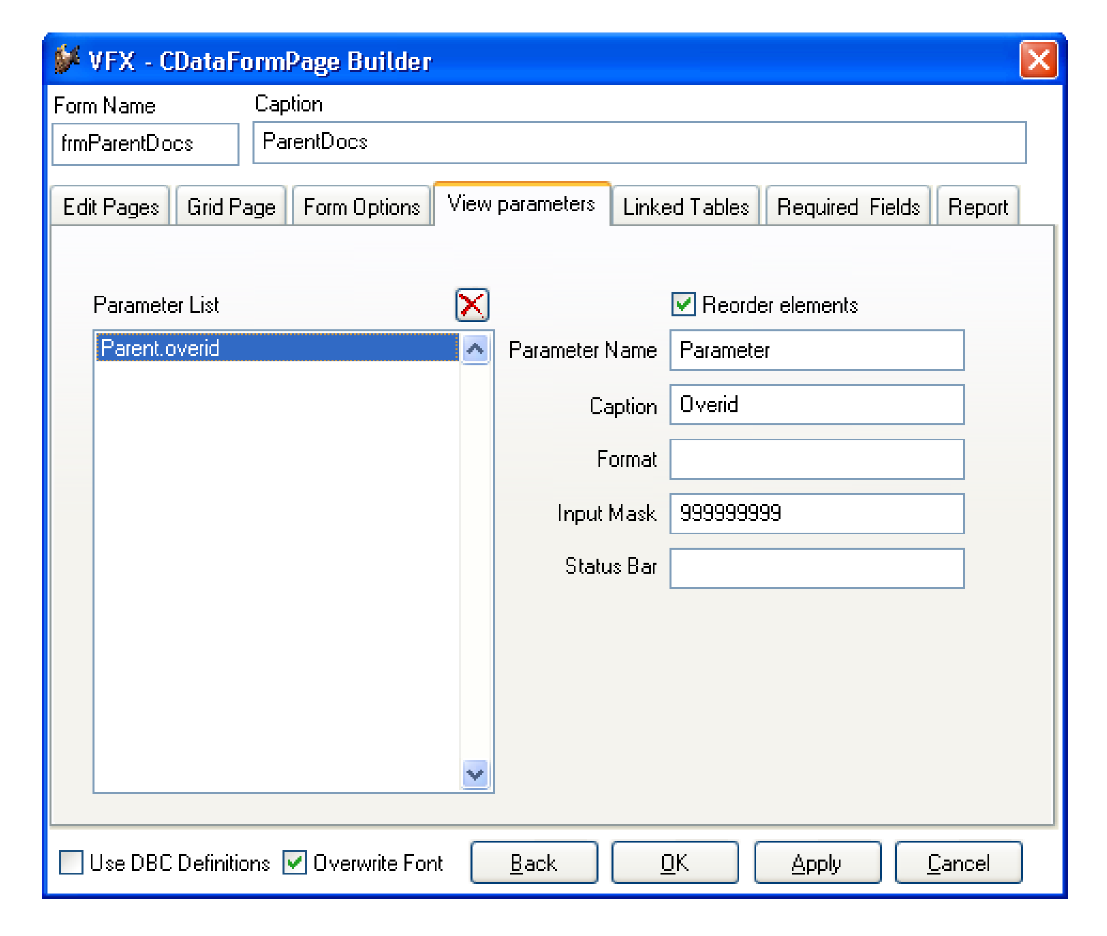
Close with ESC key. Hier wird die Formulareigenschaft *lCloseonEsc* eingestellt, die angibt, ob der Benutzer ein Formular mit der Escape-Taste schließen kann.

Save/Restore positions. Hier wird die Formulareigenschaft *lSavePosition* eingestellt, die angibt, ob die Positionen und andere Formulareinstellungen in der VFX-Ressourcentabelle gespeichert werden sollen.

Add Speedbar Control. Dieses Kontrollkästchen fügt dem Formular eine Schaltflächenleiste hinzu. Hier ein Beispiel:



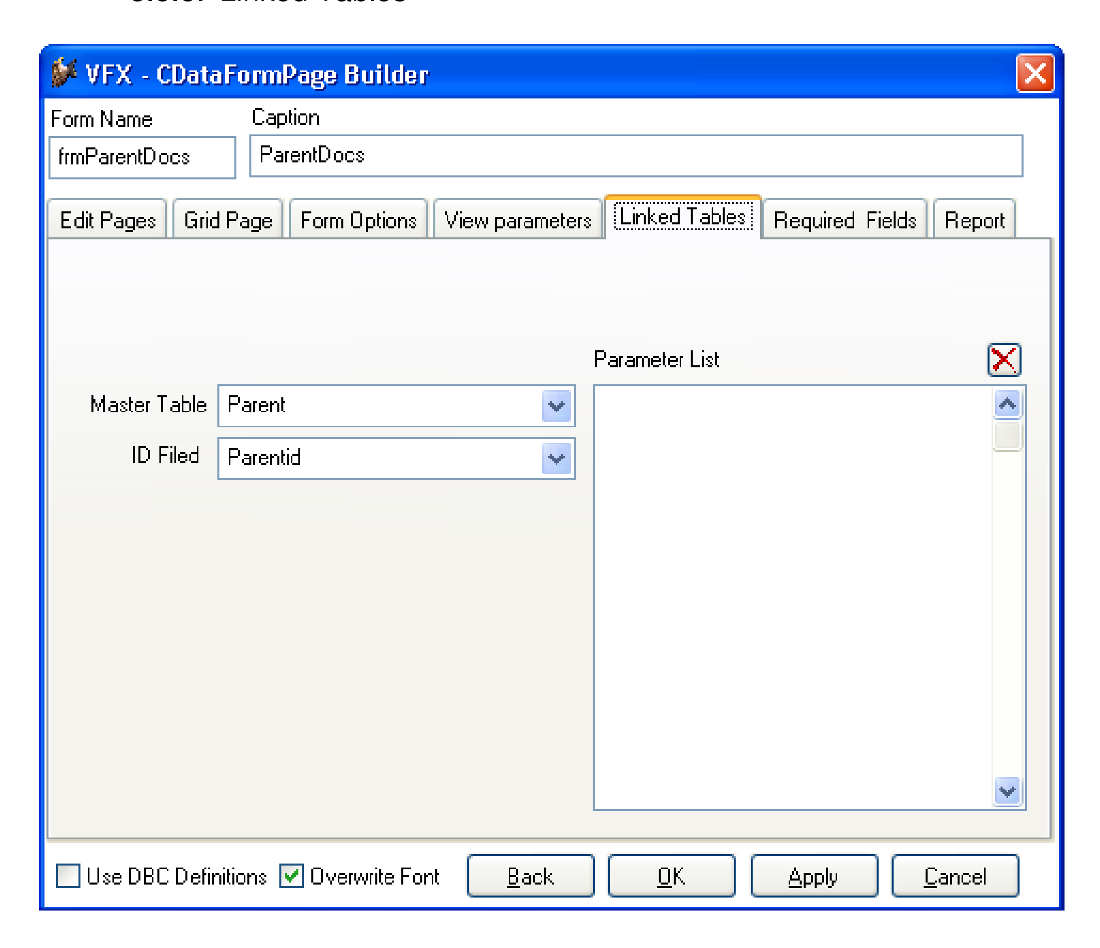
8.5.4. View Parameters



Auf der Seite *View Parameters* können Steuerelemente zur Eingabe von Ansichtsparametern angelegt werden. Ähnlich wie auf Formularen basierend auf der Klasse *cAskViewArg* kann der Benutzer hier zur Laufzeit Werte eingeben. Über eine *Requery*-Schaltfläche in der Standardsymbolleiste kann die Ansicht aktualisiert werden. Auf diesem Weg entfällt die Instanziierung eines weiteren Formulars.

Die Steuerelemente zur Eingabe von Ansichtsparametern werden am oberen Rand des Formulars, oberhalb des Seitenrahmens, platziert. Diese Steuerelemente sind immer sichtbar. Für diese Steuerelemente muss der Name eines Ansichtsparameters anstelle einer Controlsource angegeben werden.

8.5.5. Linked Tables

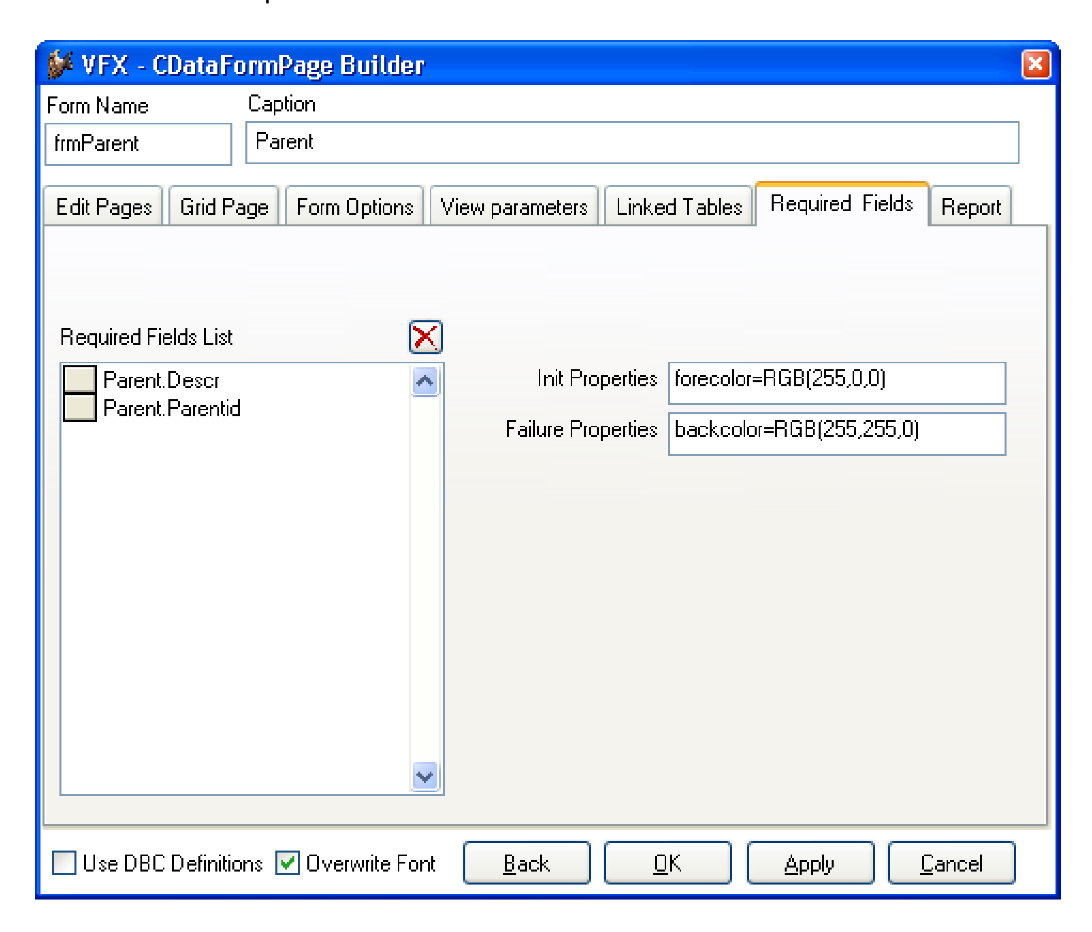


VFX 11.0-Anwendungen unterstützen 1:1-Beziehungen zwischen der Hauptbearbeitungstabelle und weiteren Tabellen. Hierdurch bekommt der Entwickler eine größere Flexibilität bei der Entwicklung komplexer Datenbanken ohne zusätzlichen Code zur Gewährung der Integrität der Datenbank schreiben zu müssen. VFX hält die Daten automatisch konsistent.

Es ist nicht notwendig, dass die Hauptbearbeitungstabelle und die in Beziehung stehenden Tabellen Primärschlüssel mit denselben Namen haben. Die Schlüsselfelder der in Beziehung stehenden Tabellen werden beim Einfügen neuer Datensätze mit dem Primärschlüssel der Haupttabelle gefüllt. Beim Löschen von Datensätzen in der Haupttabelle werden automatisch auch die in Beziehung stehenden Datensätze gelöscht.

Auf der Seite *Linked Tables* muss zunächst die Hauptbearbeitungstabelle mit dem Primärschlüssel ausgewählt werden. In der Parameterliste können Felder aus in Beziehung stehenden Tabellen gewählt werden. Es kann genau ein Feld je Tabelle ausgewählt werden. Über die ausgewählten Felder wird die Beziehung hergestellt und die referenzielle Integrität gewährleistet. Wenn versucht wird ein zweites Feld aus einer Tabelle auszuwählen, so wird das zuerst gewählte Feld überschrieben.

8.5.6. Required Fields



Mithilfe der neuen Formulareigenschaften *cRequiredFields*, *cRequiredFieldInitProps*, *cRequiredFieldFailureProps* und *cRequiredFieldFailureForm* kann verhindert werden, dass Feldinhalte mit Nullwerten oder ohne Inhalt gespeichert werden.

Der Listbox *Required Fields List* kann eine beliebige Anzahl von Datenfeldern aus dem Feldassistenten zugewiesen werden. Während der Initialisierung des Formulars werden alle Steuerelemente auf eine Controlsource aus dieser Liste überprüft. Alle Steuerelemente mit einer entsprechenden Controlsource werden als erforderliche Eingabefelder behandelt.

Die Liste der erforderlichen Eingabefelder wird vom Form Builder der Formulareigenschaft *cRequiredFields* zugewiesen.

In der Textbox *Init Properties* kann eine Semikolon-Separierte Liste mit Zuweisungen an Eigenschaften in der Form

```
PropertyName = cExpression [; PropertyName = cExpression ]
```

eingegeben werden. Für alle erforderlichen Eingabefelder werden während der Initialisierung diese Zuweisungen ausgeführt. Im Beispiel aus der Abbildung bekommen alle Steuerelemente, die ein erforderliches Datenfeld als Controlsource haben, die Vordergrundfarbe rot.

Wenn mehreren Eigenschaften Werte zugewiesen werden sollen, werden die Zuweisungen durch Semikolon getrennt. Wenn beispielsweise alle erforderlichen Eingabefelder mit einer fetten Schrift, roten Schriftfarbe und einem hellgelben Hintergrund angezeigt werden sollen, ist im Feld *Init Properties* folgender Wert einzutragen:

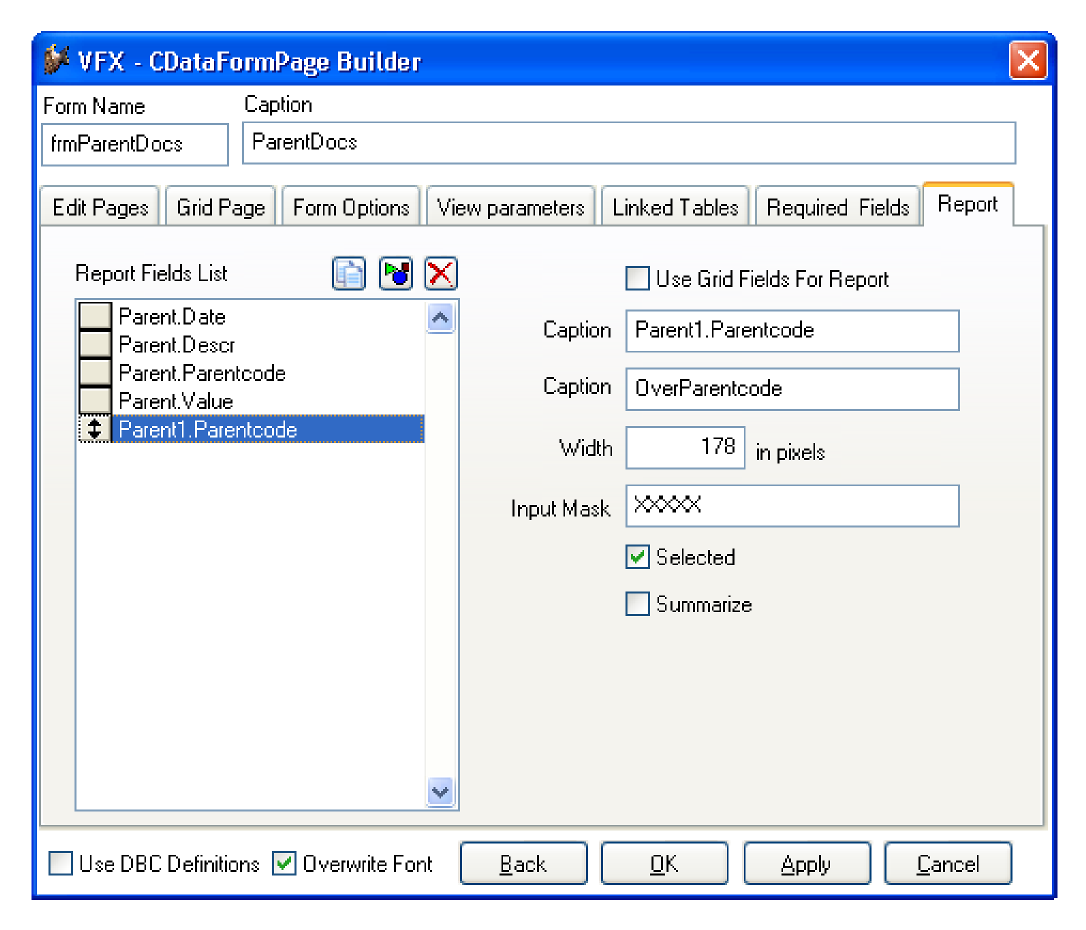
```
FontBold = .T.; ForeColor = RGB(255,0,0); BackColor = RGB(255,255,196)
```

Auf diesem Weg kann dem Benutzer auf einfachem Weg gezeigt werden, welche Felder ausgefüllt werden müssen. Der Wert des Feldes *Init Properties* wird der Formulareigenschaft *cRequiredFieldInitProps* zugewiesen.

Beim Speichern der Daten des Formulars werden alle erforderlichen Eingabefelder auf einen eingegebenen Wert überprüft. Wenn ein fehlender Wert festgestellt wird, werden dem entsprechenden Steuerelement die Eigenschaften aus dem Feld *Failure Properties* zugewiesen. Die Eingabe erfolgt nach den gleichen Regeln, wie beim Feld *Init Properties*. Der Wert des Feldes *Failure Properties* wird der Formulareigenschaft *cRequiredFieldFailureProps* zugewiesen.

Solange nicht alle erforderlichen Eingabefelder mit Werten gefüllt sind, werden die Daten des Formulars nicht gespeichert.

8.5.7. Report



Häufig ist es erforderlich auf Berichten Felder zu drucken, die auf der Listenseite eines Formulars nicht zur Verfügung stehen. Genauso kann es möglich sein, dass Felder aus dem Grid nicht gedruckt werden sollen. Die Seite *Report* ermöglicht es Felder auszuwählen, die zur Laufzeit auf der Seite *Erweitert* des Druckdialogs zur Auswahl stehen sollen. Hier kann eine Vorauswahl der standardmäßig zu druckenden Felder und der Felder mit Summierung gemacht werden.

Für jedes Feld können die Breite des Feldes, eine Eingabemaske und eine Überschrift vorgegeben werden.

Wenn, wie in früheren VFX-Versionen, alle Felder des Grids im Suchdialog verwendet werden sollen, muss das Kontrollkästchen *Use Grid Fields For Report* markiert werden.

OK. Wählen Sie diese Schaltfläche, um Ihr Formular generieren zu lassen. Dies dauert einige Sekunden und das Ergebnis ist ein Formular, auf dem Sie die gewünschte Anzahl von Bearbeitungsseiten mit den gewählten Feldern

auf jeder Seite haben. Wenn Sie mehr Felder gewählt haben als untereinander auf eine Seite passen, werden zwei Spalten erzeugt.

Der Formularerstellungsprozess kann mehrmals gestartet werden. Diese Eigenschaft nennt man wieder verwendbar.

ANMERKUNG: Die Eigenschaft wieder verwendbar ist zu 100% nur für Formulare verfügbar, die mit dem VFX-Formular-Builder erzeugt wurden. Um das wieder verwendbare Verhalten des Builders sicherzustellen sollten Sie immer den VFX-Formular-Builder verwenden, wenn Sie Ihrem Formular Felder hinzufügen wollen.

Ein weiterer großer Vorteil der wieder verwendbaren VFX-Formular-Builder ist die Tatsache, dass Sie Änderungen, die Sie in der Datenbank (z. B. Überschrift, Format oder Eingabemaske) durchgeführt haben, durch Aufrufen des VFX-Formular-Builders und auswählen des Kontrollkästchens *Use DBC Definitions* in das Formular übernehmen können.

Starten Sie Ihre Anwendung, wählen Sie im Öffnen-Dialog Ihr neu erstelltes Formular und starten Sie es mit einem Mausklick. Testen Sie es und prüfen Sie, wo Ihr Formular erweitert werden muss.

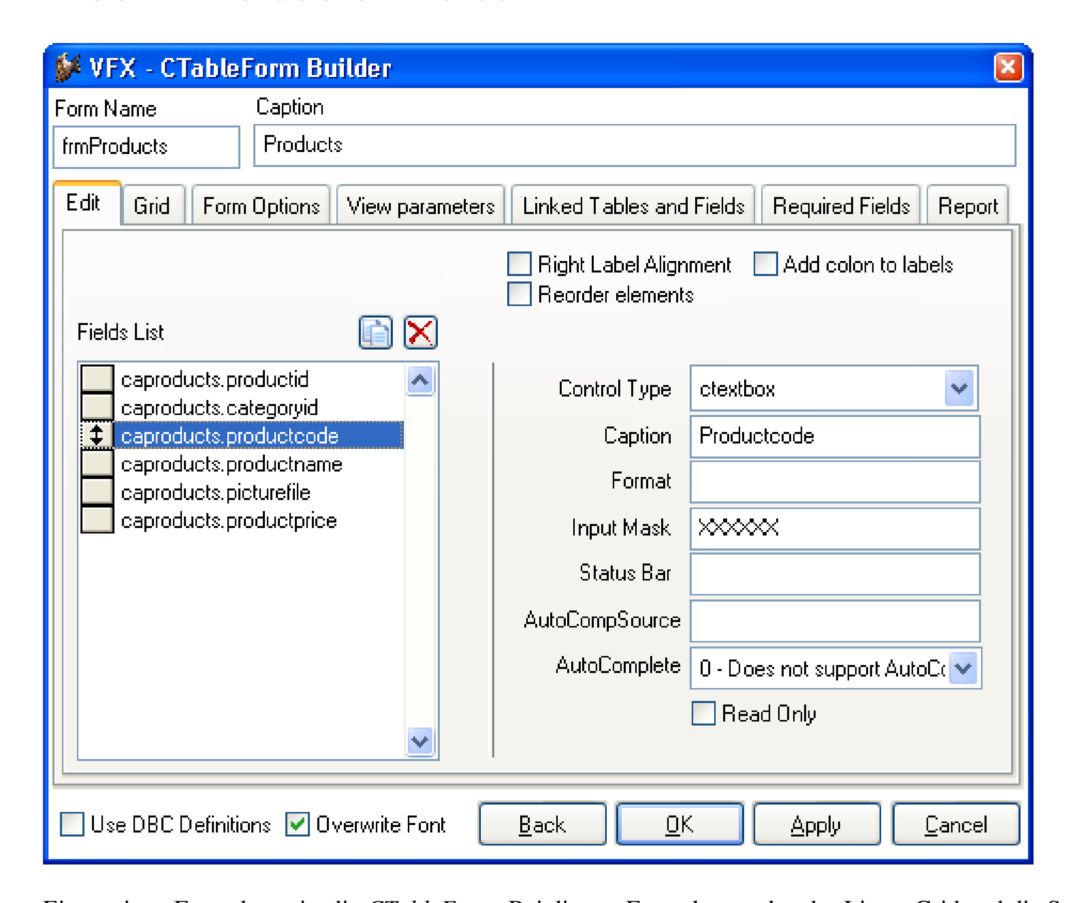
Um mit den VFX-Formularassistenten besser vertraut zu werden, lohnt es sich, einige Formulare zu generieren. Beginnen Sie mit einfachen Formularen und erweitern Sie diese später um Auswahllisten.

Nachdem Sie mit dem Erstellen von Standard-VFX-Datenbearbeitungs-Formularen vertraut sind, können Sie sich den 1:n-Datenbearbeitungs-Formularen zuwenden.

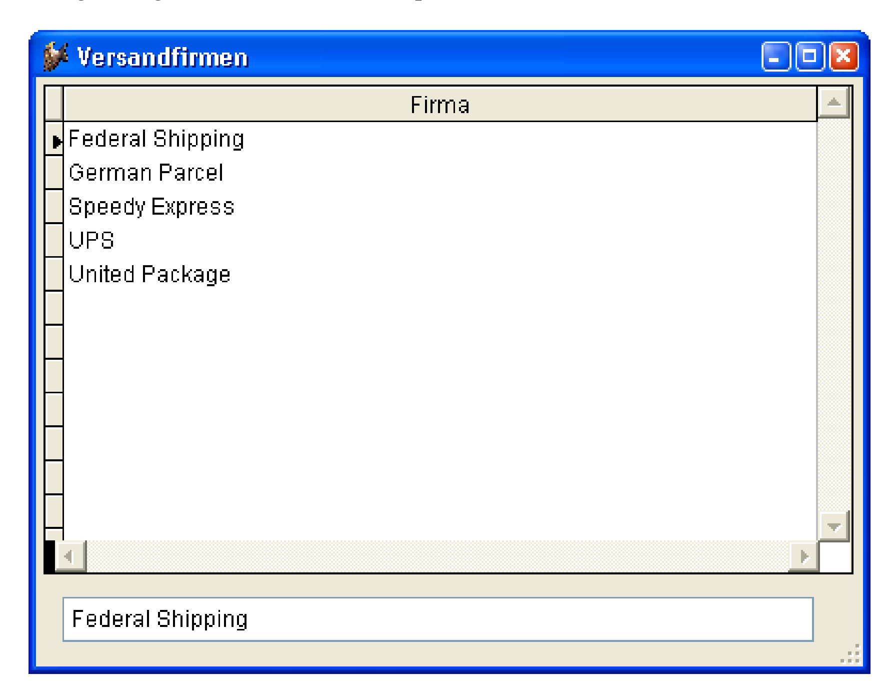
Apply. Hat die gleiche Funktion wie die Schaltfläche *OK*, schließt den VFX-Formular-Builder jedoch nicht.

Cancel. Bricht die Ausführung des VFX-Formular-Builders ab. Jede Auswahl und Eingabe geht dabei verloren.

8.6. VFX – CTableForm Builder



Eine weitere Formularart ist die *CTableForm*. Bei diesem Formular werden das Listen-Grid und die Steuer-elemente nebeneinander oder untereinander dargestellt. Es eignet sich daher insbesondere für Formulare mit nur wenigen Eingabefeldern. Hier ein Beispiel für ein Formular basierend auf der Klasse *CTableForm*:



8.7. VFX – COneToMany Builder

Das 1:n-Formular ist eine Weiterentwicklung des Standard-VFX-Datenbearbeitungs-Formulars. Das bedeutet, dass Sie auf einem einzigen Formular die normalen Datenbearbeitungsfunktionen haben können und ein Grid mit den Child-Datensätzen zu dem aktuell angezeigten Hauptdatensatz haben. VFX erlaubt es Ihnen, auch mehrere Child-Tabellen zu einer Haupttabelle auf mehreren Seiten eines Seitenrahmens zu bearbeiten. Wenn Sie viele Eingabefelder in Ihrer Child-Tabelle haben, können Sie die Felder auf mehrere Seiten eines Seitenrahmens verteilen. Das erlaubt Ihnen, eine große Anzahl verschiedenster Anwendungen abzudecken ohne wirklich programmieren zu müssen. Alles was Sie wissen müssen ist, wie man ein 1:n-Formular erstellt, die zugehörige Datenbank einrichtet und durch welche Felder die Haupttabelle und die Child-Tabelle miteinander verbunden sind. Lassen Sie uns ein einfaches Beispiel betrachten:

Wie schon weiter oben in diesem Handbuch beschrieben, müssen Sie die Datenbank Ihrer Anwendung einrichten. Definieren Sie Ihre Tabellen, Felder und Indexschlüssel sowie die Feldüberschriften. Die VFX-Builder benutzen diese Informationen, sodass Sie die Überschriften nicht nochmals eingeben müssen.

Bevor Sie ein 1:n-Formular erstellen, sollten Sie die Grundlagen des Datenbank-Designs und insbesondere 1:n-Beziehungen beherrschen. In 1:n-Beziehungen stellen Sie die Verbindung von einem Hauptdatensatz zu den Child-Datensätzen her. Ein gutes Beispiel für eine 1:n-Beziehung ist die Verbindung zwischen Aufträgen (Haupttabelle) und Auftragspositionen (Child-Tabelle) in jedem Auftragsbearbeitungssystem.

Wenn Sie die referenzielle Integrität (RI) nicht manuell mit Hilfe der VFX-Methoden wie *OnPostDelete()* herstellen wollen, ist es sinnvoll, den RI-Code im Datenbank-Designer anzulegen, bevor Sie mit der Erstellung von 1:n-Formularen beginnen. Wenn Sie diese Arbeit manuell erledigen wollen, müssen Sie den Code für das Löschen von Hauptdatensätzen und den zugehörigen Child-Datensätzen von Hand schreiben. Wenn Sie außerdem die Änderung des Schlüsselfeldes in der Haupttabelle erlauben, müssen Sie auch den Code schreiben, um die Child-Datensätze zu aktualisieren.

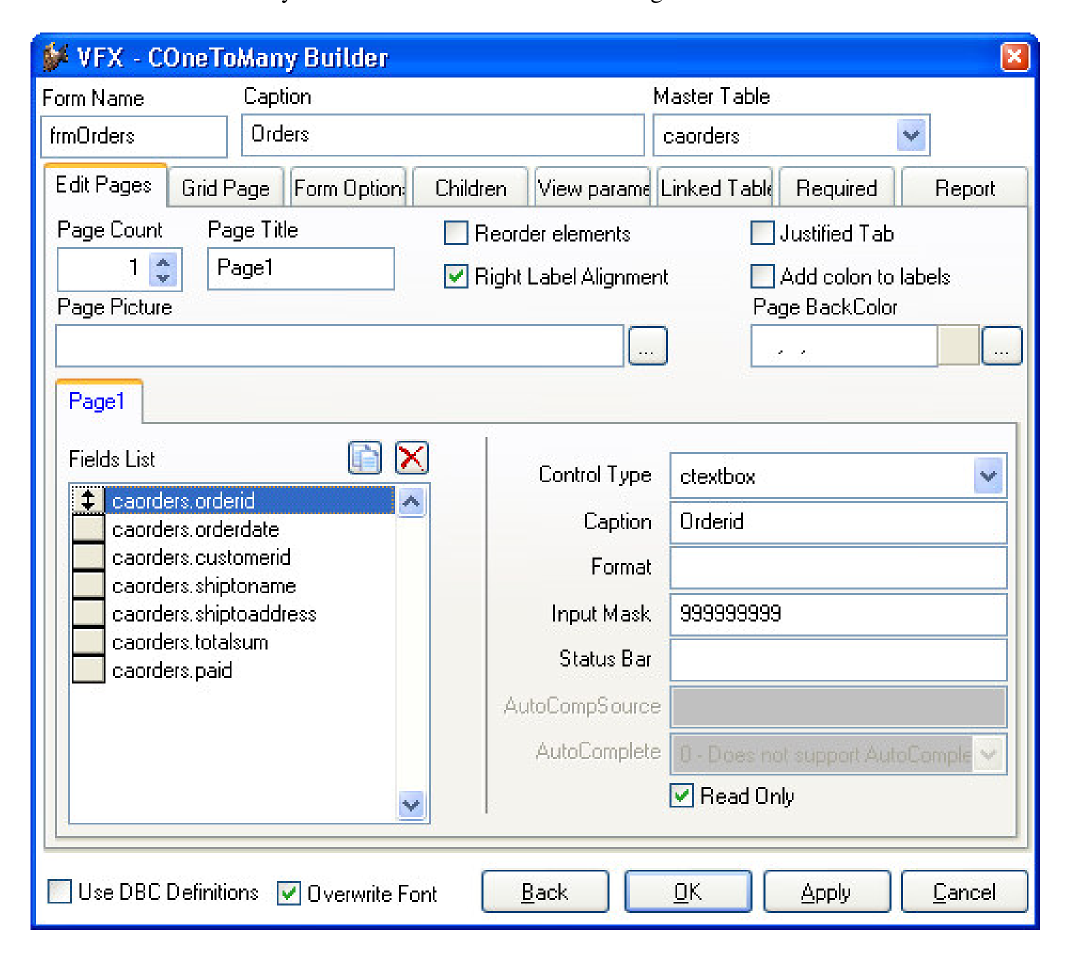
Starten Sie aus dem VFX-Menü den VFX – Form Wizard und erstellen Sie ein Formular basierend auf der Klasse *cOneToMany*.

Richten Sie mit dem VFX – Dataenvironment Builder die Datenumgebung des Formulars ein, das Sie erstellen wollen. Der VFX – COneToMany Builder verwendet diese Informationen automatisch beim Erstellen des 1:n-Formulars.

Der VFX – COneToMany Builder hilft Ihnen bei der Erstellung von anspruchsvollen 1:n-Formularen, ohne zu programmieren. Wenn Sie die 1:n-Beziehung zwischen der Haupttabelle und der Child-Tabelle hergestellt haben, können Sie 1:n-Formulare genauso einfach erstellen wie Standard-VFX-Datenbearbeitungsformulare. Wenn Sie mehrere Child-Tabellen mit einer Haupttabelle verbinden wollen, müssen Sie von jeder Child-Tabelle eine Beziehung zu der Haupttabelle herstellen.

WICHTIG: Denken Sie daran, den *InitialSelectedAlias* in der Datenumgebung anzugeben. Außerdem müssen Sie die 1:n-Beziehung zwischen der Haupttabelle und der Child-Tabelle herstellen. Ansonsten wird Ihr Formular nicht so funktionieren, wie Sie es erwarten!

Der VFX – COneToMany Builder hat eine intuitive Bedienung.



Bearbeiten Sie zunächst die folgenden Optionen:

Form Name. Geben Sie den Namen des neuen Formulars ein. Der VFX – Form Wizard hat bereits einen Standardnamen entsprechend den Namenskonventionen zugewiesen. Der Name beginnt mit *frm*. Selbstverständlich können Sie Ihrem Formular einen beliebigen Namen geben, aber wir empfehlen Ihnen, sich an die allgemeinen Namenskonventionen zu halten.

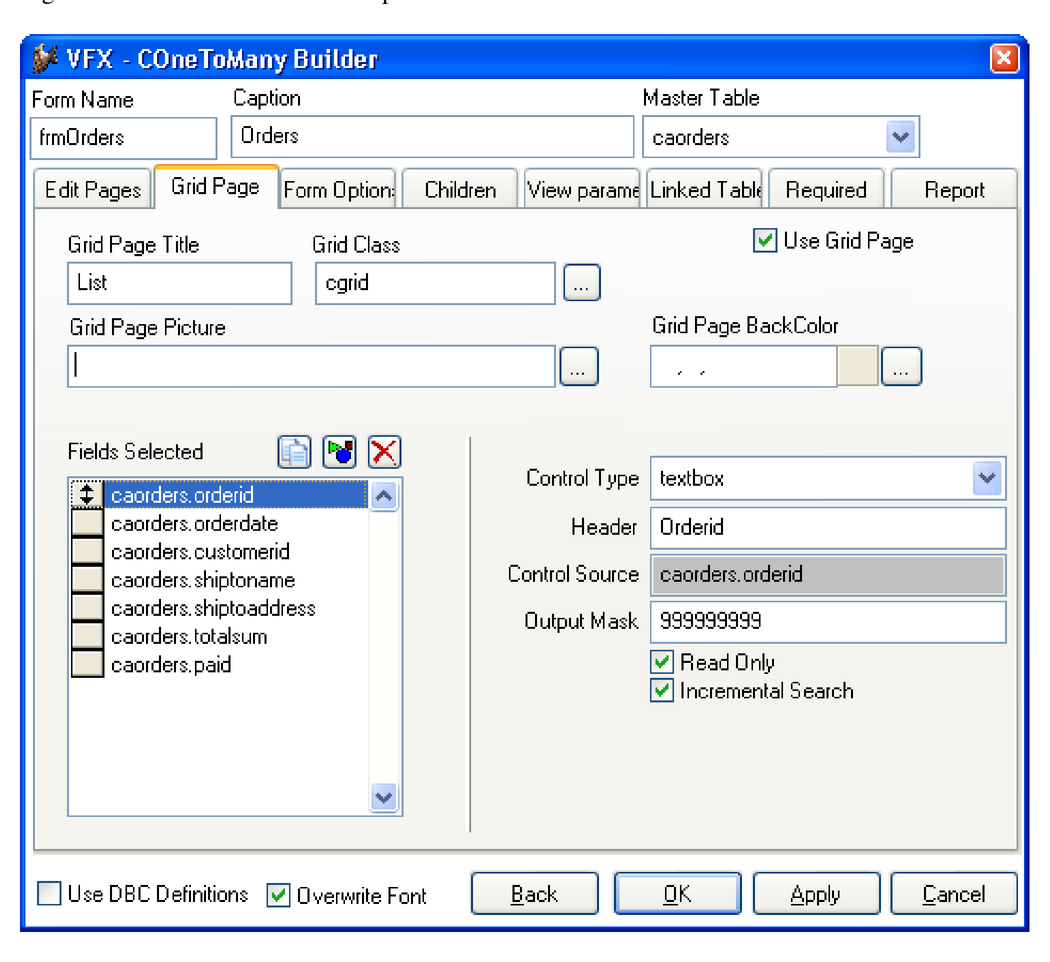
Caption. Geben Sie die Überschrift für Ihr Formular ein. Während Sie die Überschrift eingeben, wird diese bereits in der Überschrift des Formular-Builders angezeigt.

Master Table. Name der Haupttabelle oder Ansicht.

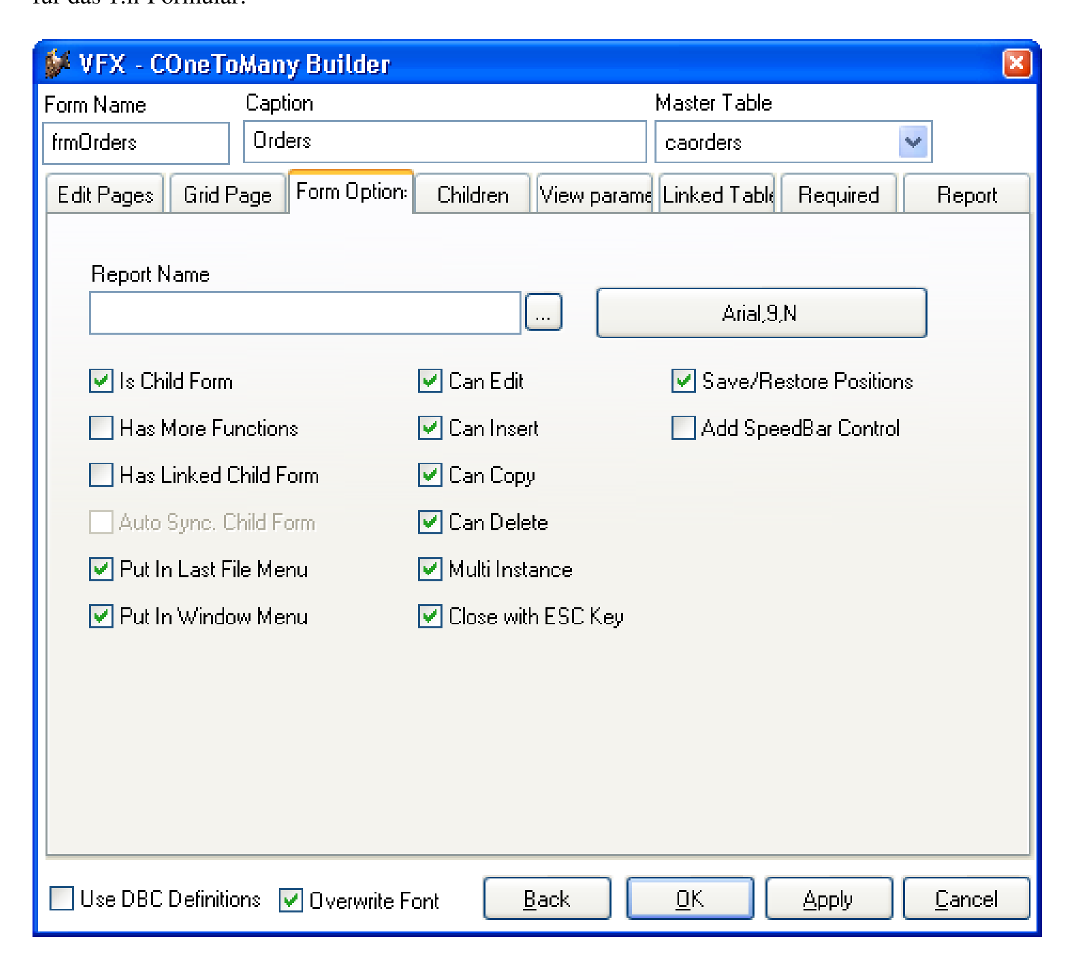
Als nächstes bearbeiten Sie den Seitenrahmen mit den Seiten *Edit Pages*, *Grid Page, Form Options* und *Children*:

Auf der Seite mit dem Namen *Edit Pages* sehen Sie die gleichen Bedienungselemente wie im VFX – CDataFormPage Builder, der weiter oben in diesem Handbuch beschrieben wurde. Hier legen Sie die Eigenschaften der Bearbeitungsseiten für die Haupttabelle fest:

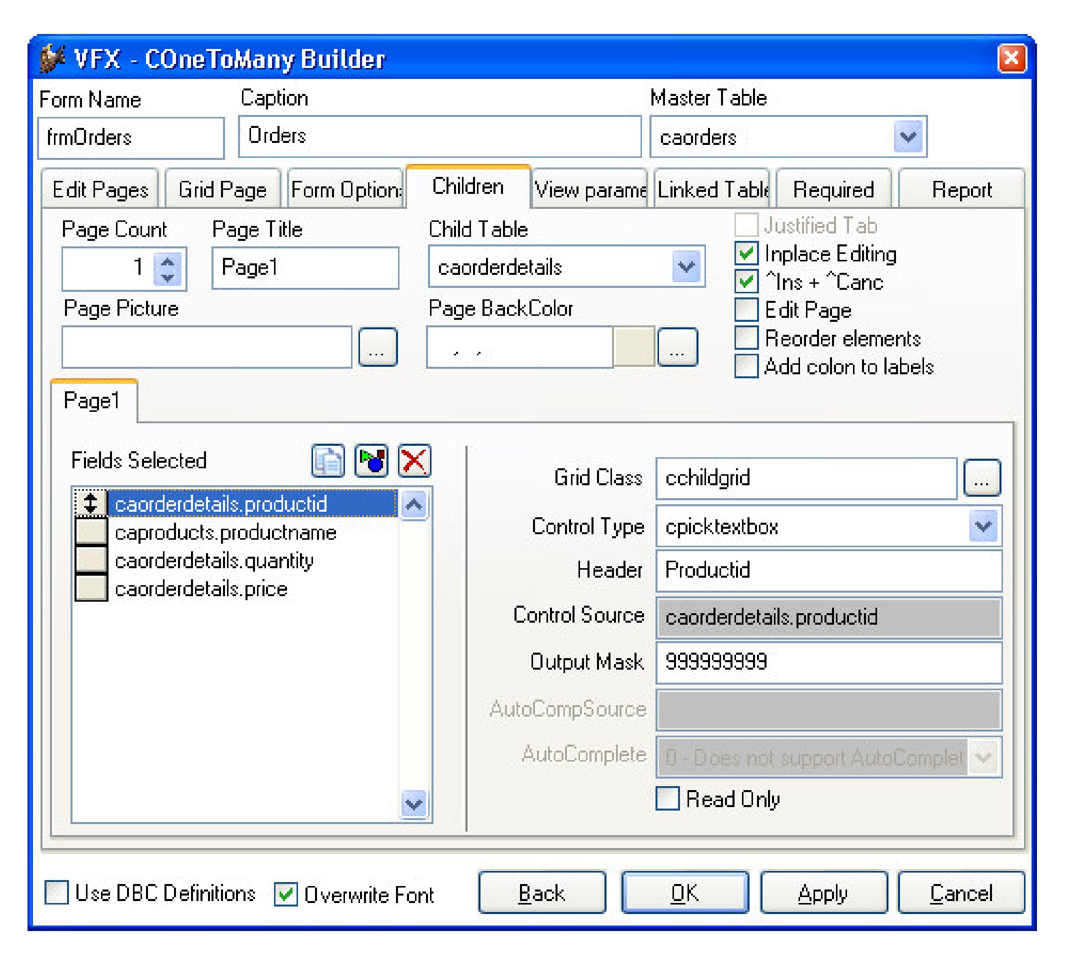
Auf der Seite mit dem Namen *Grid Page* sehen Sie die gleichen Bedienungselemente wie im VFX – CDataFormPage Builder, der weiter oben in diesem Handbuch beschrieben wurde. Hier beschreiben Sie die Eigenschaften des Grids für die Haupttabelle:



Auf der Seite mit dem Namen *Form Options* sehen Sie die gleichen Bedienungselemente wie im VFX – CDataFormPage Builder, der weiter oben in diesem Handbuch beschrieben wurde. Hier wählen Sie die Optionen für das 1:n-Formular:



Auf der Seite mit dem Namen *Children* geben Sie an, wie Child-Seiten gestaltet werden soll. Child-Seiten können wahlweise ein Grid oder andere Steuerelemente enthalten:



Page Count. Geben Sie ein, wie viele Child-Grids Ihr Formular haben soll. Für die meisten 1:n-Formulare wird ein Grid ausreichen. Wenn Sie mehrere Child-Tabellen haben, werden Sie diese über mehrere Seiten verteilen wollen. Entsprechend der Anzahl der Seiten, die Sie gewählt haben, erscheint der Seitenrahmen des Formular-Builders mit der gewählten Anzahl von Seiten. Wenn Sie zwei Seiten einstellen, hat der Seitenrahmen zwei Seiten, wenn Sie drei Seiten einstellen, hat der Seitenrahmen drei Seiten usw.

Page Title. Geben Sie die Überschrift für das aktuell gewählte Child-Grid an. Wenn Sie die Überschrift für die zweite Seite eingeben wollen, drücken Sie auf die zweite Seite. Der VFX – COneToMany Builder zeigt sofort den eingegebenen Text als Überschrift der jeweiligen Seite an.

Child Table. Geben Sie die Datenquelle für Ihr Child-Grid an. Achtung: Es ist sehr wichtig, diese Einstellung zu machen. Wenn Sie diese Eigenschaft nicht einstellen, wird Ihr Formular nicht richtig funktionieren.

Justified Tab. Markieren Sie dieses Kontrollkästchen, wenn die Seitenüberschriften justiert sein sollen. Ansonsten haben die Überschriften eine variable Länge und füllen nicht die Breite des Seitenrahmens.

Inplace Editing. Markieren Sie diese Option, wenn Sie Daten in das Child-Grid eingeben wollen, was normalerweise der Fall ist.

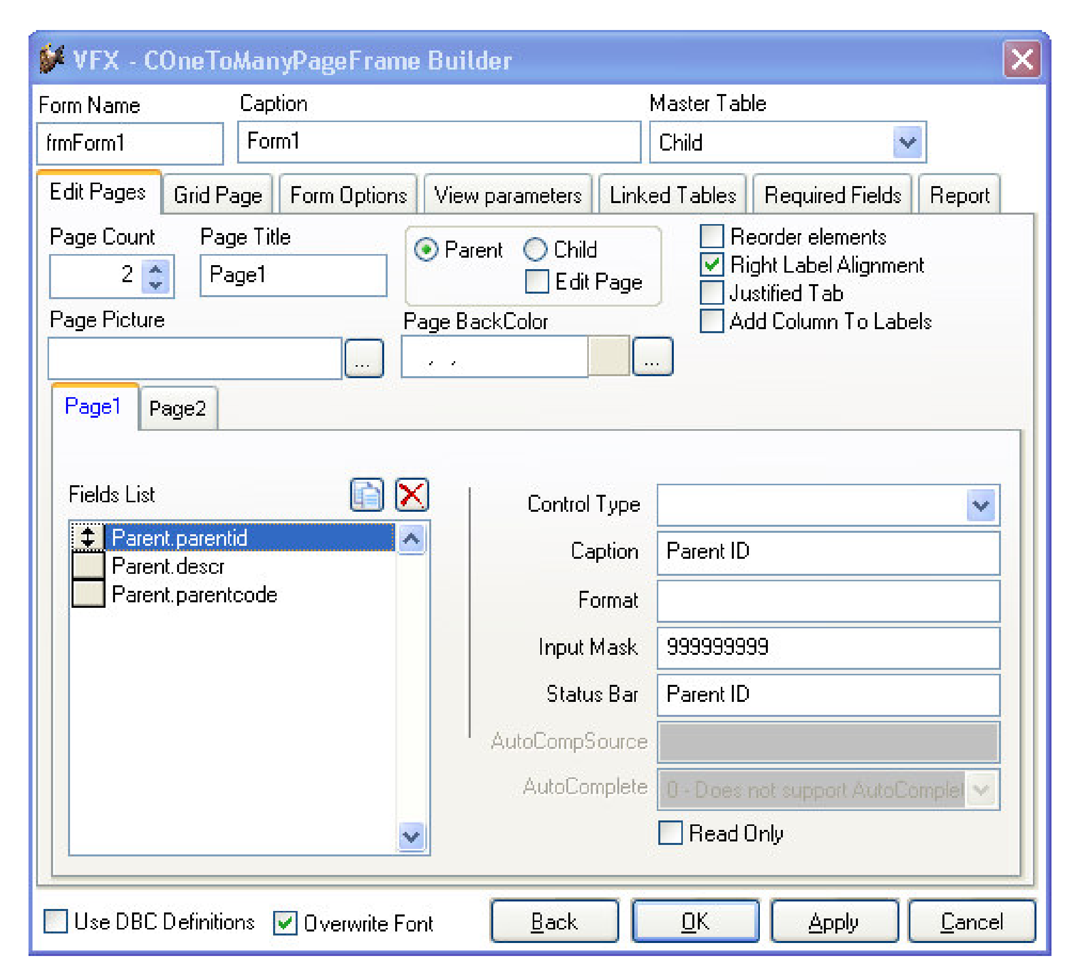
^Ins+^Canc. Markieren Sie diese Option, wenn Sie die Möglichkeit haben wollen, mit Strg+Einfg Datensätze einzufügen und mit Strg+Entf Datensätze im Child-Grid zu löschen.

Die anderen Optionen sind mit denen auf der Grid-Seite des VFX – CDataFormPage Builder identisch.

8.8. VFX – ConeToManyPageFrame Builder

Die Klasse *COneToManyPageframe* gibt dem Entwickler die Möglichkeit auf einem Seitenrahmen auf verschiedenen Seiten Parent-Daten und Child-Daten darzustellen. Die Klasse vereint die Vorteile der Klasse *CDataFormPage* mit der Möglichkeit Child-Daten zu bearbeiten.

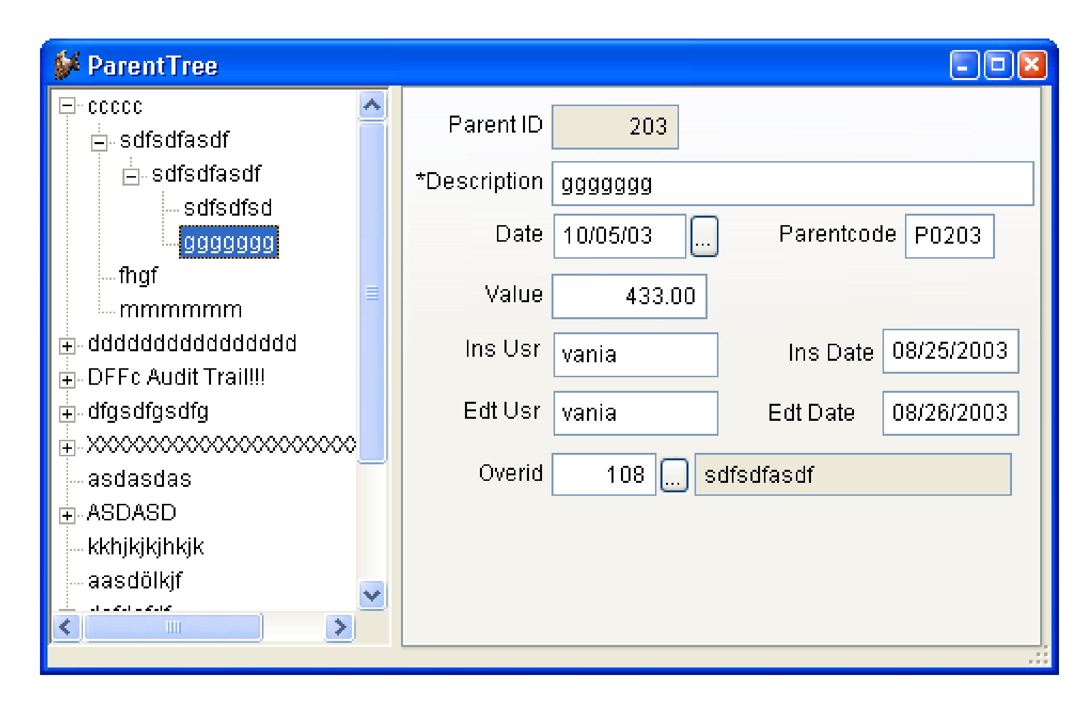
Wenn die aktive Seite des Seitenrahmens Steuerelemente vom Typ „Parent“ enthält, bezieht sich die Navigation auf die Parent-Daten. Wenn die aktive Seite des Seitenrahmens Steuerelemente vom Typ „Child“ enthält, bezieht sich die Navigation auf die Child-Daten. Auf Child-Seiten können wahlweise beliebige Steuerelemente oder ein Childgrid platziert werden.



Zusätzlich zu den Einstellungen, die der Entwickler im Form Builder für andere Formularklassen machen kann, ist es hier erforderlich einzustellen, ob eine Seite Parent-Daten oder Child-Daten enthalten soll. Wenn eine Seite eine Child-Seite sein soll, kann eingestellt werden, ob sich Steuerelemente oder ein Child-Grid auf dieser Seite befinden soll.

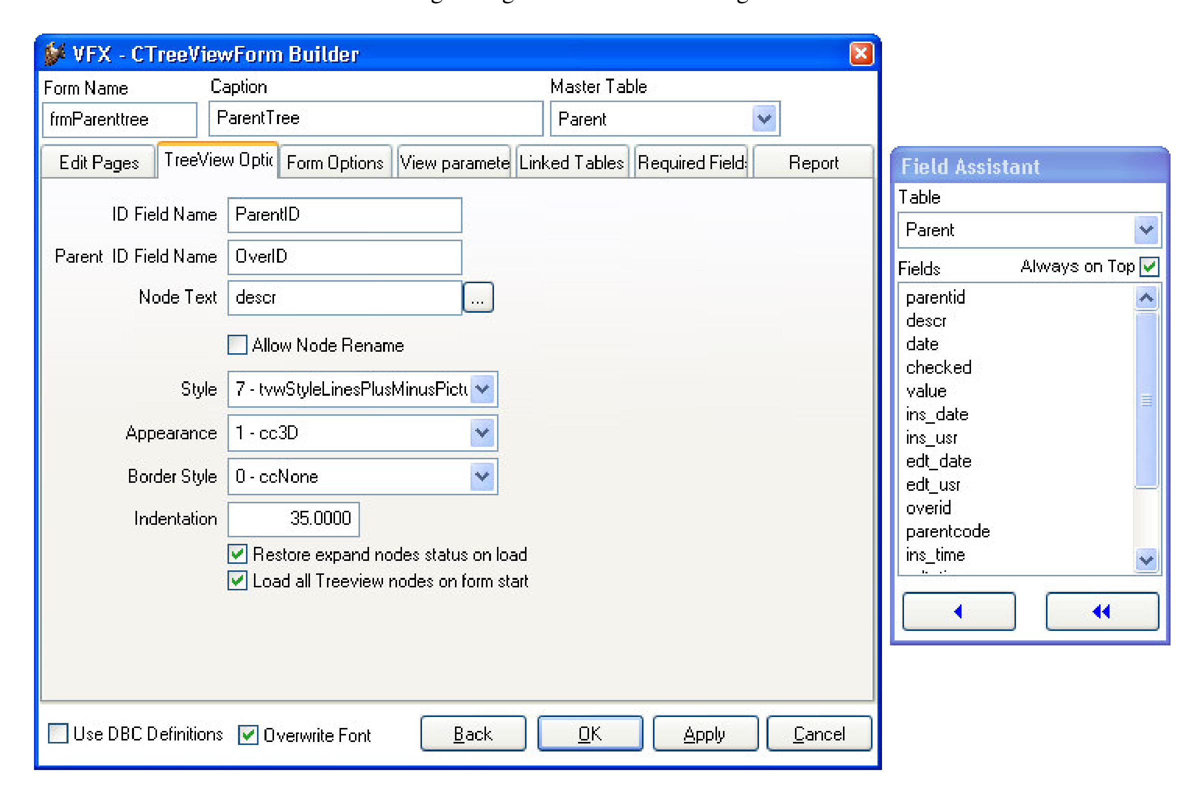
8.9. VFX – CTreeViewForm Builder

Der Haupteinsatzzweck dieser Klasse ist die Darstellung von Daten aus einer Tabelle in einer Baumstruktur. Die Baumstruktur gibt dem Endanwender einen kompletten Überblick über die hierarchischen Beziehungen in einer Tabelle. Hier ein Beispiel:



Diese Klasse basiert auf der Klasse *CDataFormPage* (*Vfxform.vcx*) und enthält ein Treeview-Steuerelement aus der Klasse *CTreeView* (*Vfxappl.vcx*). Die Klasse kombiniert die Funktionalität von *CDataFormPage* mit den Möglichkeiten der hierarchischen Datenpräsentation in einer Baumstruktur. Wenn ein Eintrag im Treeview-Steuerelement ausgewählt wird, wird der Datensatzzeiger in der zugrunde liegenden Tabelle mitgeführt und der Anwender kann die Daten im rechten Teil des Formulars bearbeiten.

Mit dem VFX – CTreeViewForm Builder können sehr schnell Formulare basierend auf der Klasse *CTreeViewForm* erstellt und alle benötigten Eigenschaften können eingestellt werden.



Der Builder arbeitet ähnlich dem VFX – CDataFormPage Builder. Die Einstellungen können auf den Seiten Edit Pages und Form Options genauso gemacht werden, wie im VFX – CDataFormPage Builder. Zusätzlich müssen

die Einstellungen für das Treeview-Steuerelement auf der Seite TreeView Options gemacht werden. Es müssen zwei Arten von Einstellungen für das Treeview-Steuerelement gemacht werden.

8.9.1. Datenanbindung des TreeView-Steuerelements

IDFieldName – Hier wird der Name des Feldes mit dem Primärschlüssel der Bearbeitungstabelle eingetragen.

ParentIDFieldName – Diese Eigenschaft enthält den Namen des Feldes, in dem der Primärschlüssel des Parent-Datensatzes gespeichert ist.

NodeText – Hier kann entweder der Name eines Feldes, das einen Beschreibungstext enthält eintragen werden oder es wird ein Ausdruck eingetragen, der zur Laufzeit evaluiert wird und dessen Rückgabewert als Bezeichnung in der Baumstruktur angezeigt wird. Wenn ein Feldname verwendet wird, kann dem Anwender erlaubt werden die Bezeichnung direkt im Treeview-Steuerelement zu ändern. Dies hängt vom Wert der Eigenschaft *AllowNodeRename* ab. Wenn *AllowNodeRename* auf .T. gesetzt ist, kann der Anwender die Bezeichnungen im Treeview-Steuerelement ändern. Dabei werden die Daten im zugrunde liegenden Tabellenfeld automatisch aktualisiert.

AllowNodeRename – Über diese Eigenschaft wird gesteuert, ob der Anwender die Bezeichnung im Treeview-Steuerelement ändern kann. Die Bearbeitung der Bezeichnung im Treeview-Steuerelement ist nur möglich, wenn die Bezeichnung auf einem einzelnen Tabellenfeld basiert. Dieses Tabellenfeld wird bei der Bearbeitung automatisch aktualisiert.

Weitere Eigenschaften:

lLoadAllTreeviewNodes – Wenn der Wert dieser Eigenschaft auf .T. eingestellt ist, werden alle Knoten des Treeview beim Laden des Formulars geladen. Wenn der Wert dieser Eigenschaft auf .F. eingestellt ist, werden beim Laden des Formulars nur die sichtbaren Knoten geladen. In diesem Fall werden beim Öffnen eines Knotens dynamisch die Untereinträge geladen.

lRestoreTreeviewStatus – Wenn der Wert dieser Eigenschaft auf .T. eingestellt ist, wird die Liste der geöffneten Knoten beim Schließen des Formulars für den angemeldeten Benutzer in der Ressourcentabelle gespeichert. Beim nächsten Laden des Formulars wird das Treeview dementsprechend wiederhergestellt.

8.9.2. Layout-Einstellungen des TreeView-Steuerelements

Diese Einstellungen entsprechen denen des TreeView-ActiveX-Steuerelements.

Style: 0 - tvwStyleText
1 - tvwStylePictureText
2 - tvwStylePlusMinusText
3 - tvwStylePlusMinusPictureText
4 - tvwStyleLinesText
5 - tvwStyleLinesPictureText
6 - tvwStyleLinesPlusMinusText
7 - tvwStyleLinesPlusMinusPictureText

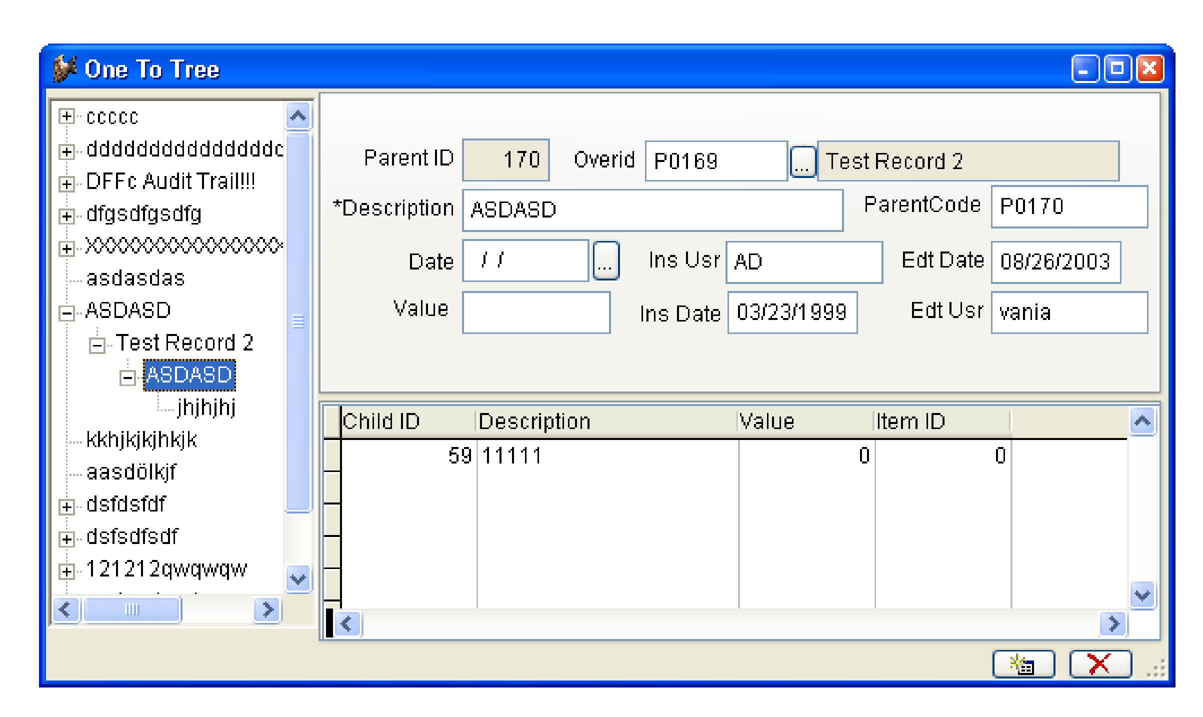
Appearance: 0 - ccFlat
1 - cc3D

BorderStyle: 0 - ccNone
1 - ccFixedSingle

Indentation: Diese Eigenschaft bestimmt die Breite des Einzugs der Knoten.

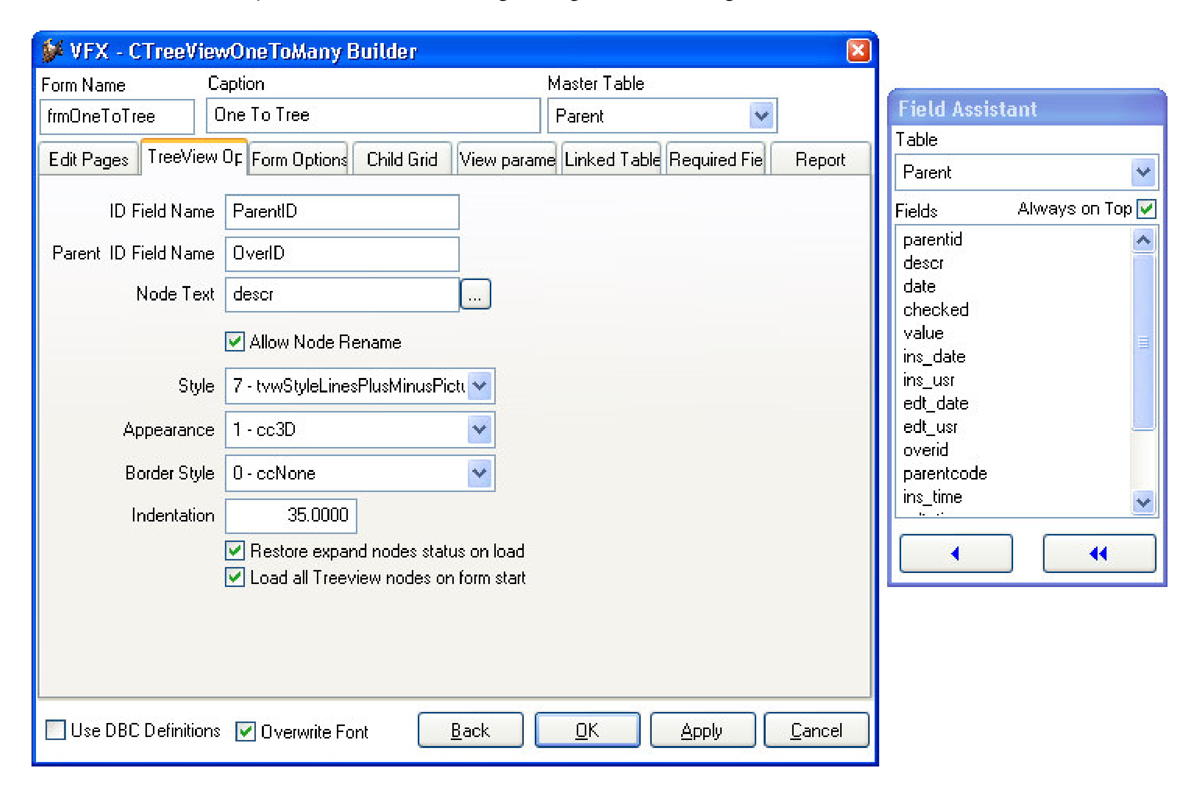
8.10. VFX – CTreeViewOneToMany Builder

Der Haupteinsatzzweck dieser Klasse ist die Darstellung der Daten aus einer Tabelle in einer Baumstruktur zusammen mit der leistungsfähigen Funktionalität, die die *COneToMany*-Klasse dem Entwickler bietet. Die Baumstruktur gibt dem Anwender den kompletten Überblick über die hierarchischen Datenbeziehungen. Hier ein Beispiel für ein Formular basierend auf der Klasse *CTreeViewOneToMany*:



Diese Klasse basiert auf der Klasse *COneToMany* (*Vfxform.vcx*) und enthält ein Treeview-Steuerelement aus der Klasse *CTreeView* (*Vfxappl.vcx*). Die Klasse kombiniert die Funktionalität von *COneToMany* mit den Möglichkeiten der hierarchischen Datenpräsentation in einer Baumstruktur. Wenn ein Eintrag im Treeview-Steuerelement ausgewählt wird, wird der Datensatzzeiger in der zugrunde liegenden Tabelle mitgeführt und der Anwender kann die Daten im rechten Teil des Formulars bearbeiten. Zusätzlich können die Child-Daten im unteren Teil des Formulars bearbeitet werden.

Mit dem VFX – CTreeViewOneToMany Builder können sehr schnell Formulare basierend auf der Klasse *CTreeViewOneToMany* erstellt und alle benötigten Eigenschaften eingestellt werden.



Dieser Builder arbeitet so ähnlich wie der VFX – COneToMany Builder. Die Einstellungen auf den Seiten Edit Pages, Form Options und Child Grid werden genauso gemacht, wie bei Formularen basierend auf der Klasse *COneToMany*. Zusätzlich müssen die Einstellungen für das Treeview-Steuerelement auf der Seite TreeView Options gemacht werden.

Die Einstellungen erfolgen genauso wie beim VFX –CTreeViewForm Builder.

8.10.1. Datenanbindung des TreeView-Steuerelements

IDFieldName – Hier wird der Name des Feldes mit dem Primärschlüssel der Bearbeitungstabelle eingetragen.

ParentIDFieldName – Diese Eigenschaft enthält den Namen des Feldes, in dem der Primärschlüssel des Parent-Datensatzes gespeichert ist.

NodeText – Hier kann entweder der Name eines Feldes, das einen Beschreibungstext enthält eintragen werden oder es wird ein Ausdruck eingetragen, der zur Laufzeit evaluiert wird und dessen Rückgabewert als Bezeichnung in der Baumstruktur angezeigt wird. Wenn ein Feldname verwendet wird, kann dem Anwender erlaubt werden die Bezeichnung direkt im Treeview-Steuerelement zu ändern. Dies hängt vom Wert der Eigenschaft *AllowNodeRename* ab. Wenn *AllowNodeRename* auf .T. gesetzt ist, kann der Anwender die Bezeichnungen im Treeview-Steuerelement ändern. Dabei werden die Daten im zugrunde liegenden Tabellenfeld automatisch aktualisiert.

AllowNodeRename – Über diese Eigenschaft wird gesteuert, ob der Anwender die Bezeichnung im Treeview-Steuerelement ändern kann. Die Bearbeitung der Bezeichnung im Treeview-Steuerelement ist nur möglich, wenn die Bezeichnung auf einem einzelnen Tabellenfeld basiert. Dieses Tabellenfeld wird bei der Bearbeitung automatisch aktualisiert.

8.10.2. Layout-Einstellungen des TreeView-Steuerelements

Diese Einstellungen entsprechen denen des TreeView-ActiveX-Steuerelements.

Style: 0 - tvwStyleText
1 - tvwStylePictureText
2 - tvwStylePlusMinusText
3 - tvwStylePlusMinusPictureText
4 - tvwStyleLinesText
5 - tvwStyleLinesPictureText
6 - tvwStyleLinesPlusMinusText
7 - tvwStyleLinesPlusMinusPictureText

Appearance: 0 - ccFlat
1 - cc3D

BorderStyle: 0 - ccNone
1 - ccFixedSingle

Indentation: Diese Eigenschaft bestimmt die Breite des Einzugs der Knoten.

8.11. Erweiterungen in OneToMany-Formularen

Gegenüber früheren VFX-Versionen, gibt es in Formularen, basierend auf den Klassen *COnetomany* und *CTreeviewOnetomany* einige Verbesserungen.

- Die Schaltflächen zum Einfügen und Löschen von Child-Daten sind nur dann enabled, wenn sich das Formular im Bearbeitungsmodus oder im Einfügemodus befindet.
- Der Child-Teil kann jetzt auch andere Steuerelemente als nur ein Childgrid enthalten.
- Bearbeitungsseiten im Child-Teil von Onetomany-Formularen können mit dem Form Builder genauso erstellt werden, wie Bearbeitungsseiten im Parent-Teil.

Die Klasse *CChildgrid*, die auf allen OneToMany-Formularen zur Bearbeitung der Child-Daten verwendet wird, wurde um einige Funktionen erweitert.

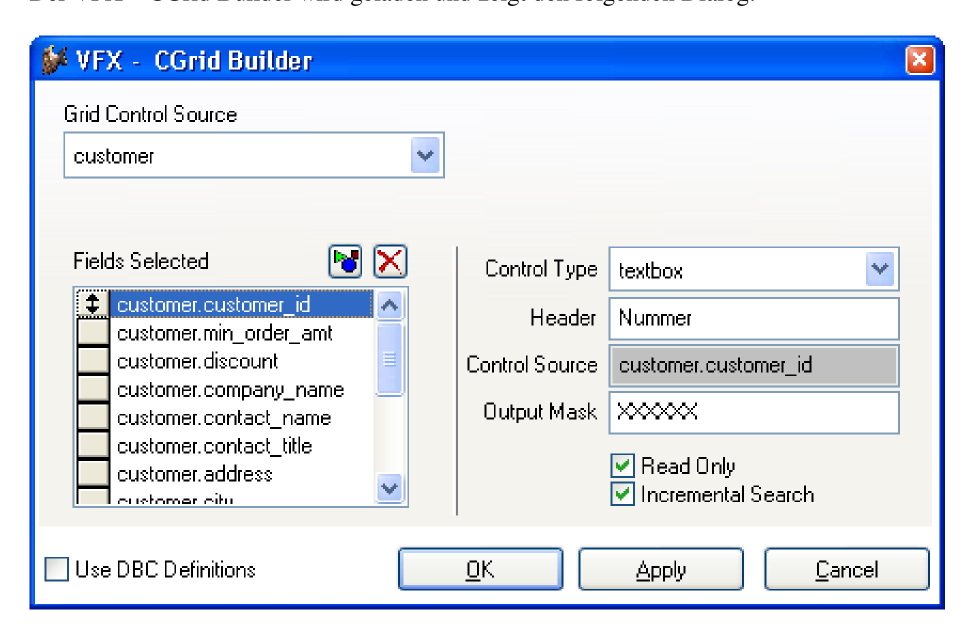
- Wenn die Child-Daten auf einer Ansicht oder auf einem CursorAdapter basieren, kann jetzt in den Child-Daten inkrementell gesucht werden.
- Ein Klick in den leeren Bereich eines Child-Grids fügt einen neuen Child-Datensatz an.

8.12. VFX – CGrid Builder

Obwohl der VFX-Formular-Builder bereits eine Seite mit einem Grid anlegt, kann es sein, dass Sie nur in diesem Grid Änderungen durchführen wollen. Der VFX – CGrid Builder automatisiert die Erstellung von leistungsfähigen Grids. Die resultierenden VFX Power Grids sind einfach zu bedienen und bringen keine Geschwindigkeitseinbußen mit sich. Sie werden die Eigenschaften der VFX Power Grids sehr nützlich finden. Die inkrementelle Suche sowie die benutzerspezifische Speicherung der Spaltenreihenfolge, Spaltenbreiten und Sortierfolge des Grids werden von den Benutzern Ihrer Anwendung geschätzt werden.

Um den VFX – CGrid Builder aufzurufen, wählen Sie die letzte Seite Ihres Formulars und wählen Sie das Grid-Steuerelement aus. Um den Builder aufzurufen, drücken Sie die rechte Maustaste und wählen Sie Builder.

Der VFX – CGrid Builder wird geladen und zeigt den folgenden Dialog:



Die Bedienung ist die gleiche wie auf der Grid-Seite des VFX-Formular-Builders. Für eine detaillierte Beschreibung aller Optionen lesen Sie bitte die Beschreibungen im Abschnitt *VFX – CDataFormPage Builder* nach.

8.13. VFX – CChildGrid Builder

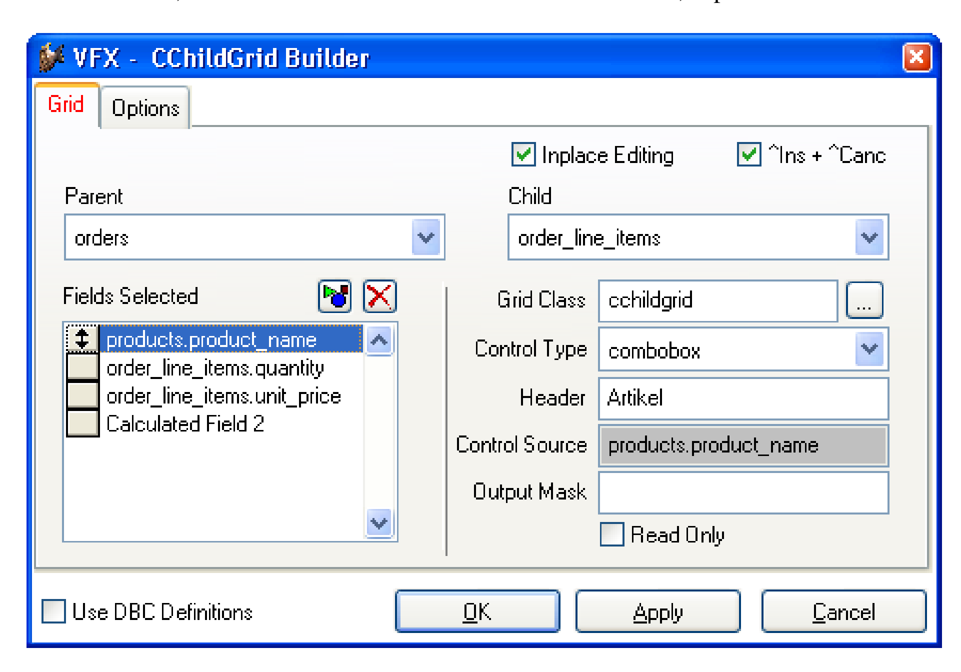
Der VFX – CChildGrid Builder erlaubt Ihnen, die Funktionalität der Child-Grids zu erweitern. Benutzen Sie diesen Builder, um die Felder für das Grid zusammenzustellen oder um den Code der Methode *OnPostInsert()* zu bearbeiten. Diese Methode wird immer dann ausgeführt, wenn dem Child-Grid ein neuer Datensatz hinzugefügt wurde. Ähnlich wie im Standard-VFX-Datenbearbeitungsformular stehen Ihnen hier die folgenden Ereignisse zur Verfügung:

- *OnPreInsert()*
- *OnInsert()*
- *OnPostInsert()*

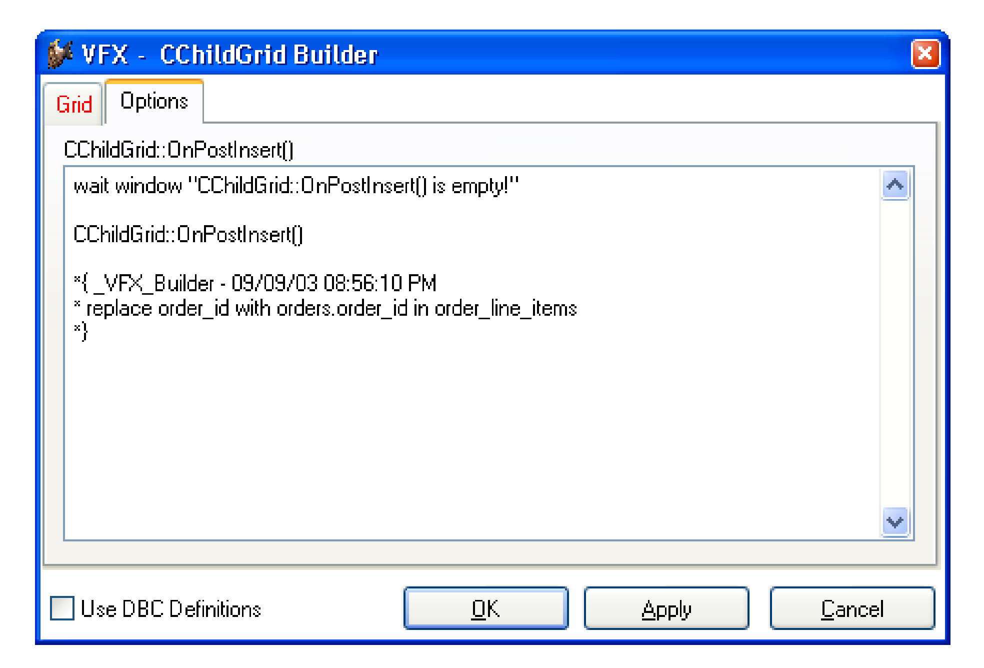
In der *OnPostInsert()*-Methode des Child-Grids müssen Sie das Feld der Child-Tabelle ausfüllen, das die Verknüpfung zur Haupttabelle herstellt. Normalerweise benötigen Sie dafür folgenden Code:

```
REPLACE <ChildLinkField> WITH <Master.MasterField> IN <ChildTable>
```

Der VFX – CChildGrid Builder ist wie folgt zu bedienen. Auf der ersten Seite mit dem Namen *Grid* können Sie das Child-Grid, wie weiter oben in diesem Abschnitt beschrieben, anpassen:



Auf der zweiten Seite mit dem Namen *Options* können Sie den Code der *OnPostInsert()*-Methode bearbeiten, um das Feld der Child-Tabelle mit dem Wert der Haupttabelle zu füllen.



Der Grund, aus dem der VFX-Builder den Code der *OnPostInsert()*-Methode nicht automatisch generieren kann, ist, dass Sie zusammengesetzte Schlüssel verwenden könnten oder mehreren Feldern in der Child-Tabelle Werte zuweisen möchten. Wenn einfache Schlüssel verwendet werden, ist der generierte Code in der Regel richtig.

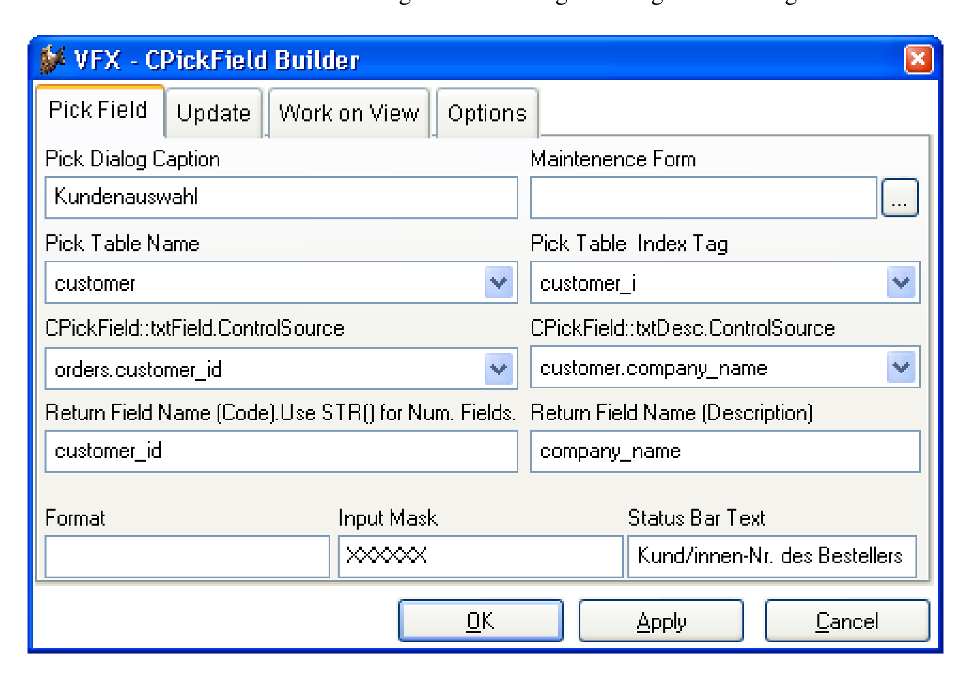
8.14. VFX – CPickField Builder

VFX enthält mehrere Klassen für Auswahlfelder. Ein Auswahlfeld besteht aus einem Textfeld, einer Schaltfläche und einem schreibgeschützten Textfeld. In dem Textfeld kann ein Wert eingetragen werden. Beim Verlassen des Feldes wird überprüft, ob der eingegebene Wert in der Tabelle mit den Auswahlwerten enthalten ist. Falls nein, wird ein Auswahlformular gestartet. Im Auswahlformular kann der Anwender den gewünschten Datensatz auswählen. In einem schreibgeschützten Textfeld können weitere Informationen aus der Auswahltabelle angezeigt werden. Auf Wunsch kann dem Benutzer erlaubt werden neue Datensätze in der Auswahltabelle zu erfassen. Alle Eigenschaften des Auswahlfeldes können mit dem VFX – CPickField Builder gemacht werden. Und das, ohne eine einzige Zeile Code oder Text im Eigenschaftsfenster des Auswahllisten-Containers manuell eintragen zu müssen!

Um den VFX – CPickField Builder aufzurufen, wählen Sie das Auswahllisten-Container-Steuerelement auf dem Formular, drücken die rechte Maustaste und wählen Builder.

ANMERKUNG: Um ein Steuerelement auszuwählen, das sich auf einer Seite in einem Seitenrahmen auf einem Formular befindet, müssen Sie den Visual FoxPro-Weg benutzen, um Steuerelemente innerhalb der Containerhierarchie auszuwählen (Klick, Rechtsklick, bearbeiten). Eine gute Möglichkeit, um festzustellen ob Sie das richtige Steuerelement ausgewählt haben, ist ein Blick in das Eigenschaftsfenster.

Der VFX – CPickField Builder wird geladen und zeigt den folgenden Dialog:



Auf der Seite *Pick Field* stehen die folgenden Optionen zur Verfügung:

Pick Dialog Caption. Geben Sie die Überschrift für das Auswahllisten-Formular ein. In diesem Formular kann der Benutzer einen Wert auswählen.

Maintenance Form. Wenn der Benutzer den gewünschten Datensatz in dem Auswahllisten-Formular nicht findet, möchten Sie dem Benutzer vielleicht die Möglichkeit geben, das normale Bearbeitungsformular aufzurufen. Geben Sie hier den Namen für das Bearbeitungsformular ein. Es wird aufgerufen, wenn der Benutzer auf die Schaltfläche *Bearbeiten…* im Auswahllisten-Formular drückt.

Pick Table Name. Wählen Sie den Namen der Tabelle oder Ansicht aus der Sie den Wert auswählen oder überprüfen möchten. Hier können Sie zwischen allen Tabellen oder Ansichten aus der Datenumgebung wählen.

Pick Table Index Tag. Dieser Indexschlüssel wird zur Überprüfung der Benutzereingabe verwendet.

CPickField::txtField.ControlSource. Dies ist die Datenquelle für das Eingabetextfeld.

CPickField::txtDesc.ControlSource. Wählen Sie die Datenquelle für das Beschreibungsfeld des Auswahllisten-Steuerelementes. Stellen Sie sicher, dass Sie eine korrekte Beziehung zu der Tabelle herstellen aus der diese Datenquelle stammt. Andernfalls wird dieses Steuerelement nicht den gewünschten Wert anzeigen, wenn Sie den Datensatzzeiger in Ihrem Formular bewegen.

Return Field Name (Code). Geben Sie den Namen des Feldes (aus der Tabelle oder Ansicht der Auswahlliste) ein, das den ausgewählten Wert enthält. Geben Sie keinen Aliasnamen ein, weil Tabellen für Auswahllisten mit einem temporären Namen geöffnet werden.

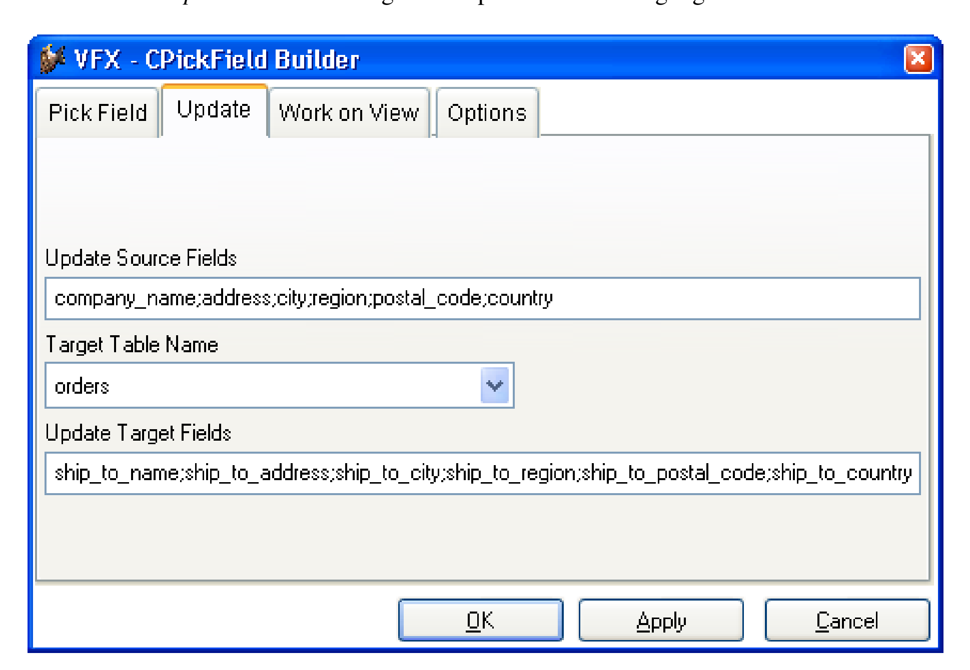
Return Field Name (Description). Geben Sie den Namen des Feldes (aus der Tabelle oder Ansicht der Auswahlliste) ein, das den Wert mit der Beschreibung enthält. Geben Sie keinen Aliasnamen ein, weil Tabellen für Auswahllisten mit einem temporären Namen geöffnet werden.

Format. Der VFX – CPickField Builder übernimmt diese Eigenschaft aus dem Datenbank-Container.

Input Mask. Der VFX – CPickField Builder übernimmt diese Eigenschaft aus dem Datenbank-Container.

Status Bar Text. Der VFX – CPickField Builder übernimmt diese Eigenschaft aus dem Datenbank-Container.

Auf der Seite *Update* stehen die folgenden Optionen zur Verfügung:

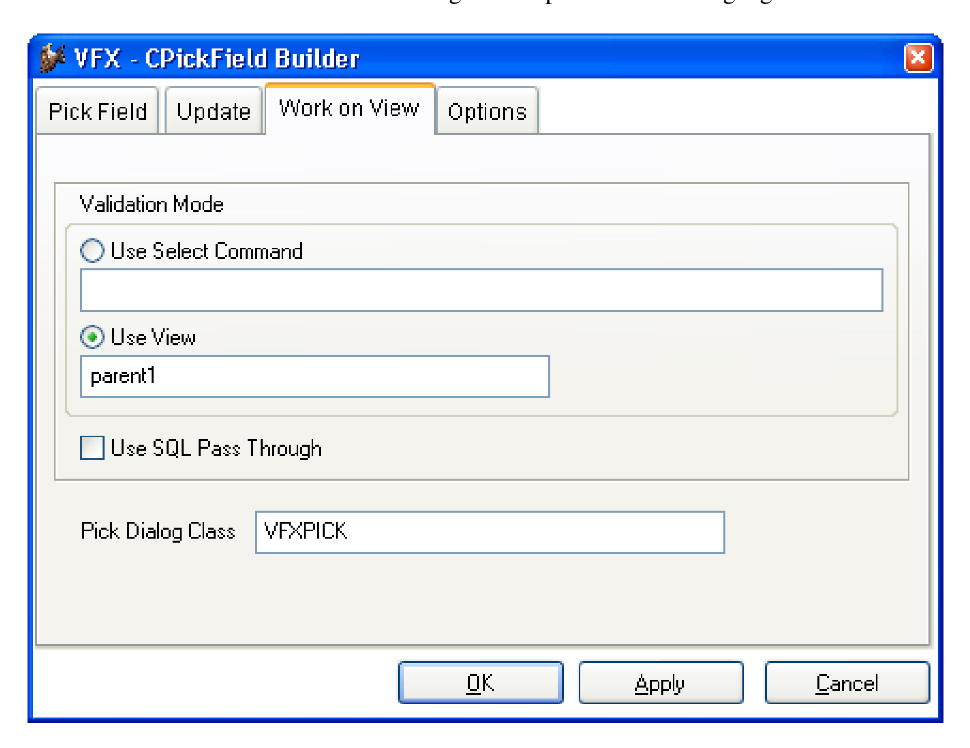


Update Source Fields. Hier können sie Felder aus der Auswahlliste eingeben, deren Werte in die Bearbeitungstabelle übernommen werden sollen. Wenn Sie mehrere Werte eingeben, so müssen diese durch Semikolon getrennt werden.

Target Table Name. Wählen sie die Zieltabelle aus. Normalerweise ist dies die Bearbeitungstabelle des Formulars.

Update Target Fields. Weisen sie die Zielfelder zu. Wenn Sie mehrere Werte eingeben, so müssen diese durch Semikolon getrennt werden.

Auf der Seite *Work on View* stehen die folgenden Optionen zur Verfügung:



Work on View. Wenn die Daten, aus denen Sie auswählen aus einer Ansicht stammen, markieren Sie dieses Kontrollkästchen.

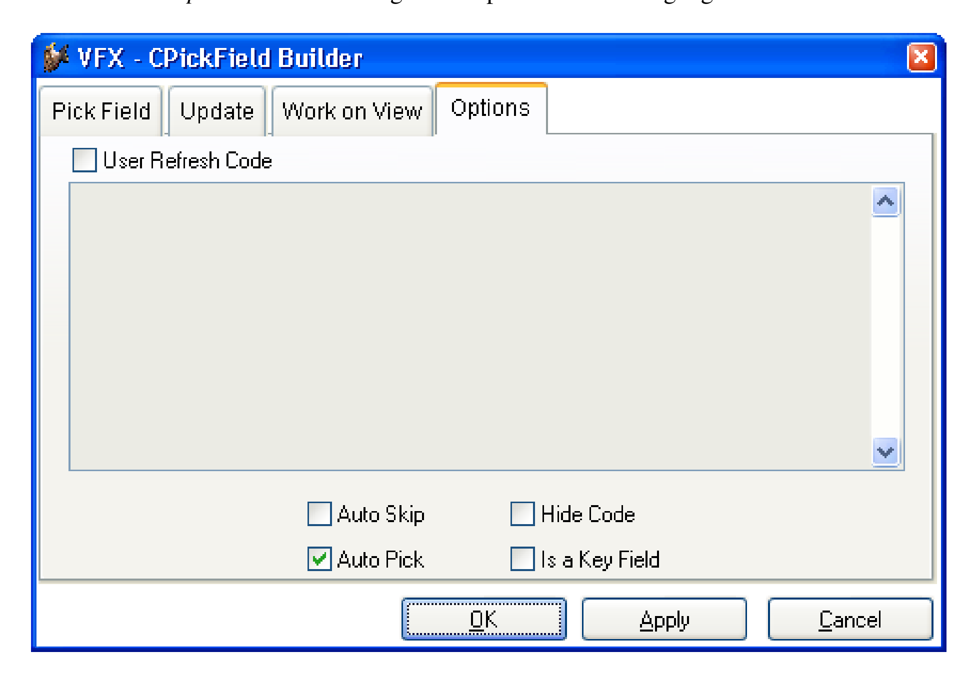
Use Select Command: Wahlweise kann ein Select-Befehl oder eine Ansicht zur Überprüfung der Benutzereingabe verwendet werden. Wenn Sie einen Select-Befehl verwenden, muss durch eine Where-Klausel sichergestellt sein, dass maximal ein Wert zurückgegeben wird. Beispiel: *„select customer_id from lv_customer where customer_id = trim(this.txtField.Value)"*

Use View: Wahlweise kann ein Select-Befehl oder eine Ansicht zur Überprüfung der Benutzereingabe verwendet werden. Wenn Sie eine Ansicht verwenden, geben Sie hier den Namen der Ansicht ein. Die Where-Klausel der Ansicht muss sicherstellen, dass maximal ein Wert zurückgegeben wird.

Use SQL Pass Through: Wenn Sie dieses Kontrollkästchen markieren, wird der in der Ansicht enthaltene Select-Befehl von VFX ausgelesen und per SQL Pass Through an die Remote-Datenquelle gesendet.

Pick Dialog Class: Hier kann eine eigene Klasse für das Auswahllisten-Steuerelement verwendet werden. Beachten Sie, dass die Klasse von der Klasse *CPickField* abgeleitet sein muss.

Auf der Seite *Options* stehen die folgenden Optionen zur Verfügung:



User Refresh Code. Manchmal benötigen Sie speziellen Code in der *Refresh()*-Methode des Auswahllisten-Containers.

Auto Skip. Markieren Sie diese Option, wenn Sie automatisch zum nächsten Steuerelement springen wollen, nachdem Sie einen Wert aus der Auswahlliste ausgewählt haben. Dadurch wird die *CPickField*-Eigenschaft *lUseTab* auf *.T.* gesetzt.

Auto Pick. Markieren Sie diese Option, wenn Sie automatisch die Auswahlliste aufrufen wollen, wenn der Benutzer einen falschen Wert eingegeben hat. Dadurch wird die *CPickField*-Eigenschaft *lAutoPick* auf *.T.* gesetzt.

Hide Code. Markieren Sie diese Option, wenn Sie das Eingabefeld in der Auswahlliste verstecken wollen. Dadurch wird die *CPickField*-Eigenschaft *lHideCode* auf *.T.* gesetzt. Der Benutzer kann keinen Wert eingeben, sondern nur aus der Auswahlliste auswählen.

Is a Key Field. Markieren Sie diese Option, wenn Sie dieses Auswahllistenfeld als Schlüsselfeld definieren wollen. Ein Schlüsselfeld ist nur zugänglich während Sie einen neuen Datensatz anlegen (so wie die Textfeld-Klasse *ckeyfield*). Dadurch wird die *CPickField*-Eigenschaft *lKeyField* auf *.T.* gesetzt.

OK. Die eingestellten Optionen werden in das ausgewählte Auswahllisten-Objekt eingefügt.

Apply. Macht das gleiche wie *OK,* jedoch wird der VFX – CPickField Builder nicht beendet.

Cancel. Bricht die Arbeit mit dem VFX – CPickField Builder ab. Alle Eingaben werden verworfen.

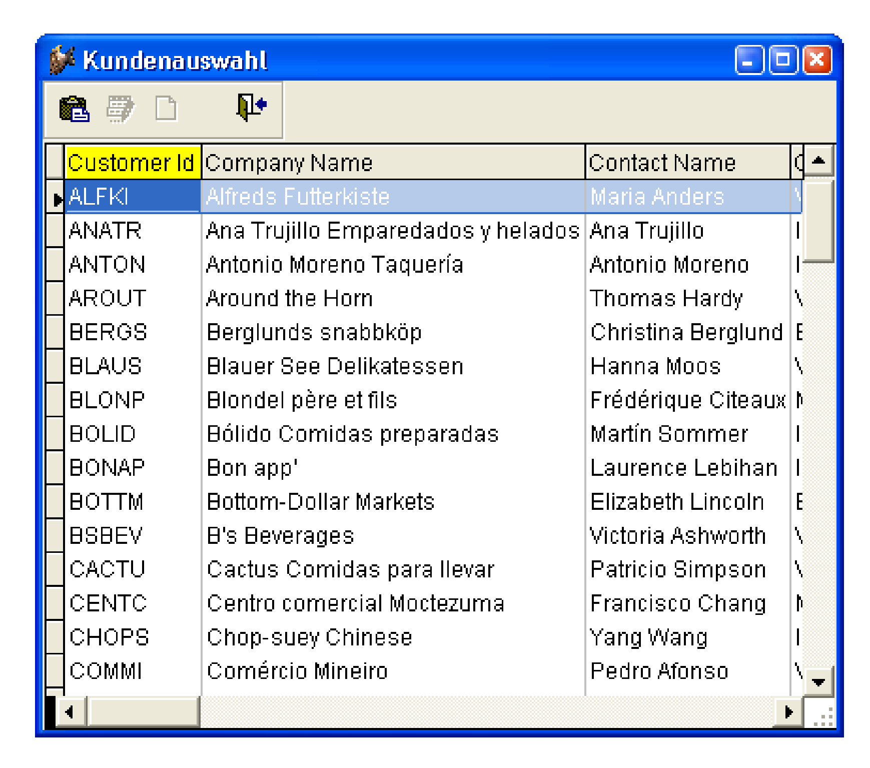
Auch dieser Builder ist voll wieder verwendbar. Das bedeutet, dass Sie diesen Builder während des Entwicklungsprozesses beliebig oft verwenden können ohne die Eigenschaften zu verlieren, die Sie bereits eingestellt haben.

Wenn Sie ein Auswahllisten-Steuerelement auf einem Formular einsetzen, sieht das etwa so aus:

Kunde: ALFKI ... Alfreds Futterkiste

Der Benutzer kann die Auswahlliste auf folgende Weise aufrufen:

- Drücken der Schaltfläche neben dem Auswahllisten-Eingabefeld (normalerweise mit drei Punkten beschriftet).
- Doppelklick auf das Auswahllisten-Eingabefeld oder auf den Beschreibungstext.
- Drücken der Funktionstaste F9.



Der Dialog der Auswahlliste hat folgende Eigenschaften (wie jedes VFX Power Grid):

- Inkrementelle Suche mit automatischer Einstellung der Sortierfolge.
- Einstellen der Sortierfolge durch Doppelklick auf die Spaltenüberschrift.
- Die Breite der Spalten kann verändert werden.
- Position und Gestaltung des Grids werden automatisch gespeichert.

Der Benutzer kann den gewünschten Datensatz auf folgende Weise auswählen:

- Doppelklick.
- Drücken der Taste *Eingabetaste*.
- Drücken der Schaltfläche *Übernehmen*.

Wenn der Benutzer die Tabelle bearbeiten möchte, die der Auswahlliste zugrunde liegt, kann er auf die Schaltfläche *Bearbeiten…* drücken. Daraufhin erscheint das Bearbeitungsformular für diese Tabelle. Wenn der Benutzer neue Datensätze hinzufügen will, drückt er auf die Schaltfläche *neu*.

8.15. VFX – CPickAlternate Builder

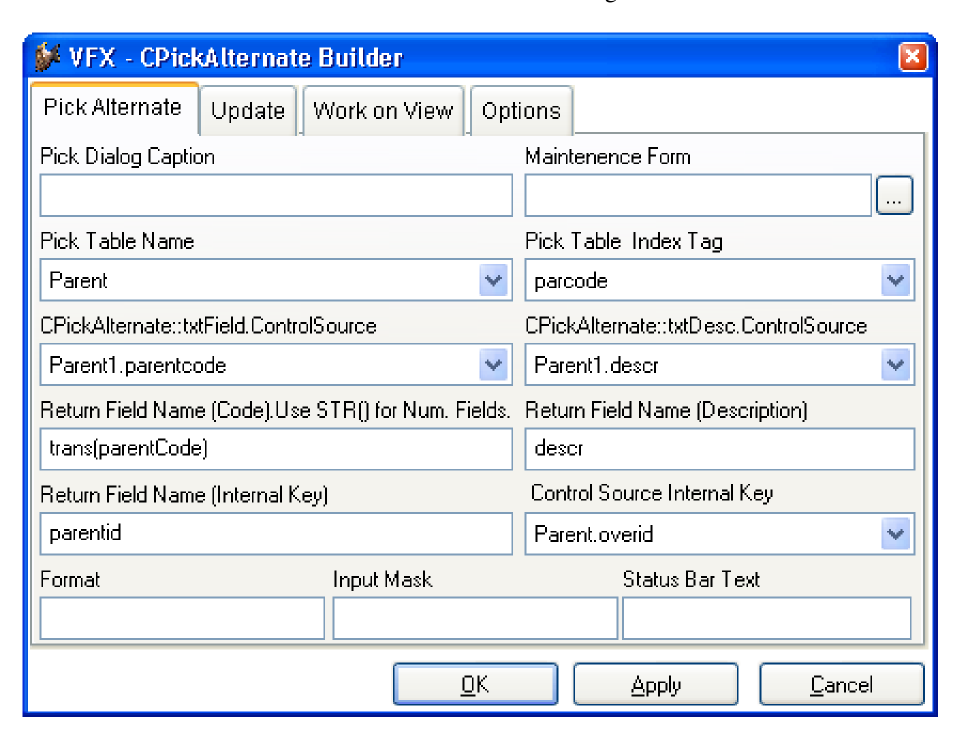
Ähnlich zum *CPickField*-Steuerelement kann die Klasse *CPickAlternate* verwendet werden um eine Benutzereingabe zu verifizieren. Es kann eine Auswahlliste aufgerufen werden, die dem Anwender erlaubt einen Wert aus einer Liste auszuwählen. Bei Verwendung der Klasse *CPickAlternate* wird der Primärschlüssel des ausgewählten Datensatzes in der Bearbeitungstabelle gespeichert während der Benutzer einen Wert aus einem anderen Feld aus der Auswahltabelle angezeigt bekommt.

Das *CPickAlternate*-Steuerelement ist einer Combobox zu bevorzugen, wenn aus einer Tabelle mit vielen Datensätzen ausgewählt werden soll. Der Einsatz ist auch sinnvoll, wenn der vom Anwender eingegebene Wert nicht dem Schlüssel der Auswahltabelle entspricht. Das Ziel dieser Klasse ist es dem Anwender eine einfach zu

bedienende Schnittstelle zu geben, die es erlaubt ihm bekannte Werte einzugeben anstelle von vom Programm generierten Primärschlüsseln. Der vom Anwender eingegebene Wert wird verwendet um den dazugehörigen Datensatz in der Auswahltabelle zu finden. Wenn der gesuchte Datensatz gefunden ist, wird als Rückgabewert der Primärschlüssel an das *CPickAlternate*-Steuerelement zurückgegeben.

Diese Klasse basiert auf der Klasse *CPickField* und erbt alle ihre Eigenschaften und Methoden. Zusätzlich hat diese Klasse die neue Eigenschaft *cControlSourceInternalKey* in die der Name des Feldes der Bearbeitungstabelle mit dem Fremdschlüssel eingetragen wird. Dieser Fremdschlüssel entspricht dem Primärschlüssel aus der Auswahltabelle.

Mithilfe des VFX – CPickAlternate Builder können die Eigenschaften dieser Klasse einfach eingestellt werden.



Pick Table Name – Hier kann der Name der Auswahltabelle aus einer der Datenquellen der Datenumgebung ausgewählt werden.

Pick Table Index Tag – Dies ist der Name des Indexschlüssels, der verwendet wird um in der Auswahltabelle zu suchen. Dieser Indexschlüssel entspricht dem Wert des Eingabefeldes.

CPickAlternate.txtField.ControlSource – Die Controlsource des Eingabefeldes. Dieses Feld muss aus der Auswahltabelle stammen.

CpickAlternate.txtDesc.ControlSource – Der Name des Beschreibungsfeldes. Der Wert wird nach der erfolgreichen Überprüfung der Benutzereingabe im Beschreibungsfeld angezeigt. Dieses Feld stammt ebenfalls aus der Auswahltabelle.

Return Field Name (Code) – Der Name des Feldes mit dem vom Anwender eingegebenen Wert aus der Auswahltabelle. In der Regel entspricht dieser Feldname dem Namen, der in *txtField.ControlSource* angegeben ist. Hier ist nur der Feldname ohne den Tabellennamen anzugeben. Der Wert dieses Feldes muss vom Typ „Zeichen“ sein. Gegebenenfalls ist der Wert mit TRANSFORM() in einen Zeichentyp umzuwandeln.

Return Field Name (Description) – Der Name des Feldes mit der Beschreibung, die aus der Auswahltabelle zurückgegeben wird. Es kann auch ein Ausdruck zurückgegeben werden. Der Wert wird im Beschreibungsfeld angezeigt. Der Wert muss vom Typ „Zeichen“ sein. Gegebenenfalls ist der Wert mit TRANSFORM() in einen Zeichentyp umzuwandeln.

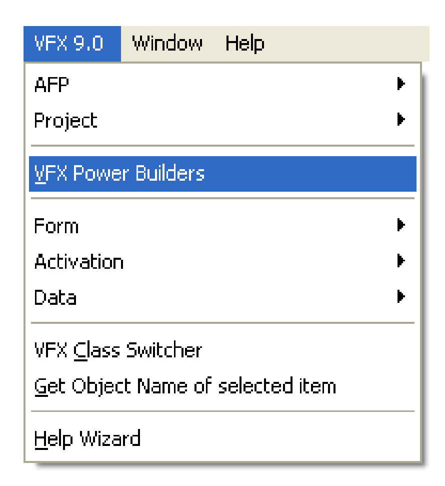
Return Field Name (Internal Key) – Der Name des Feldes aus der Auswahltabelle, das den Primärschlüssel enthält. Über dieses Feld wird die Beziehung von der Bearbeitungstabelle zur Auswahltabelle in der Datenumgebung hergestellt.

Control Source Internal Key – Der Name des Feldes aus der Bearbeitungstabelle, das den Primärschlüssel enthält. Dieses Feld enthält den Fremdschlüssel aus der Auswahltabelle.

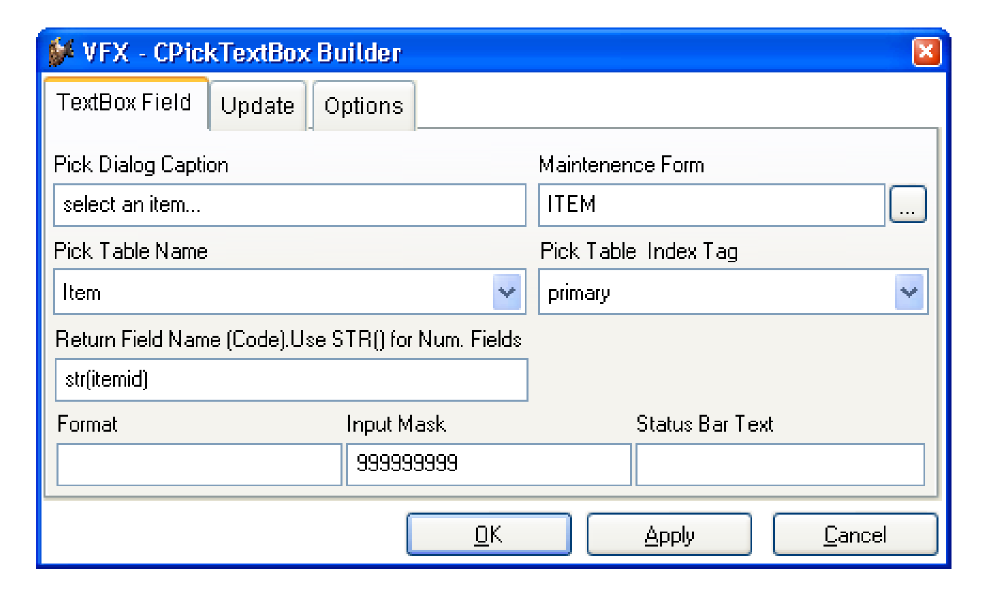
8.16. VFX – CPickTextBox Builder

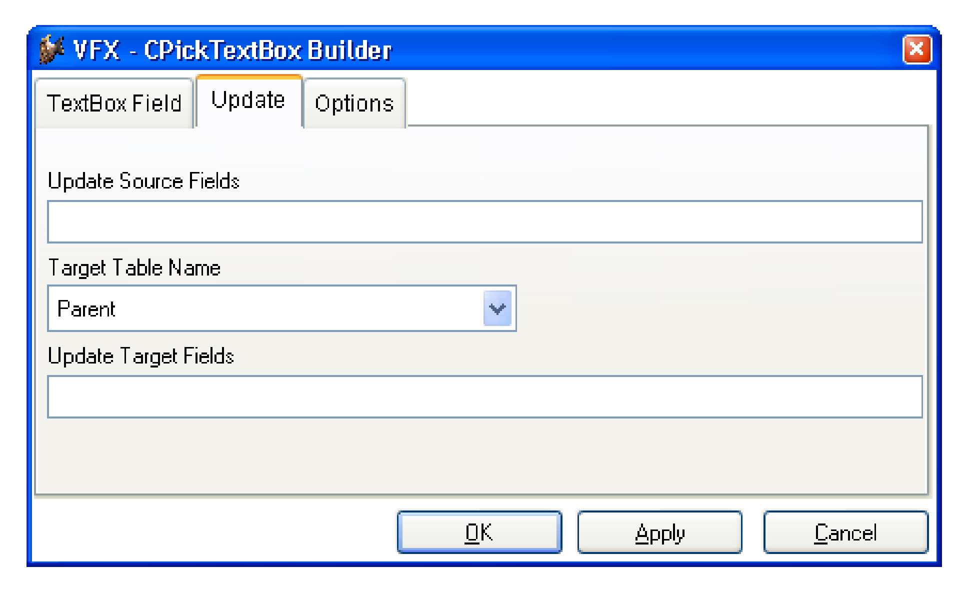
Visual Extend bietet einen Builder, um leistungsfähige Auswahltextfelder für Childgrids zu erstellen.

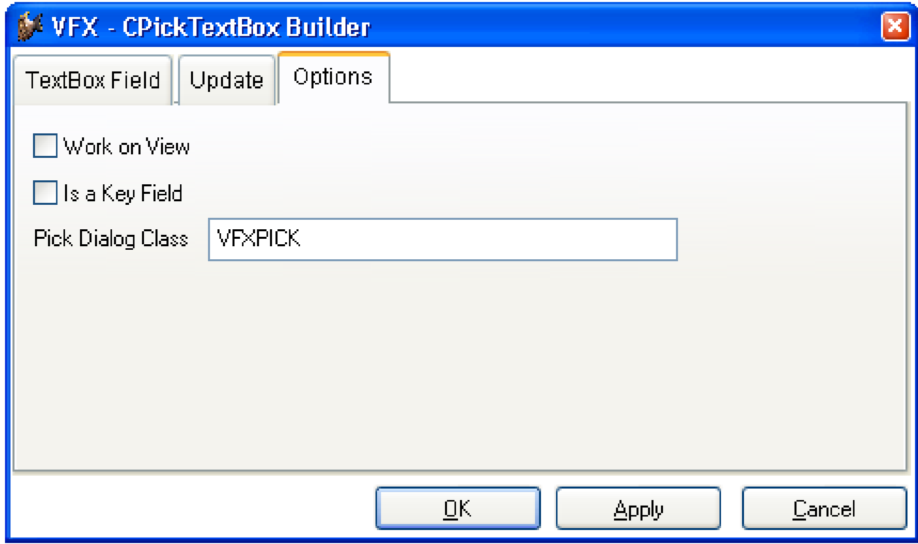
Um den VFX – CPickTextBox Builder aufzurufen, wählen Sie die Spalte im Grid, die das Auswahltextfeld erhalten soll und wählen Sie den Menüpunkt VFX Power Builder aus dem VFX-Menü:



Der VFX – CPickTextBox Builder ist in der Bedienung dem normalen VFX – CPickField Builder ähnlich und ist ebenfalls voll wieder verwendbar:







8.17. VFX – Combo Pick List Builder

Diese Klasse dient zur einfachen Erstellung von Auswahllisten. Es können Auswahllisten erstellt werden, die nicht auf einer eigenen Tabelle basieren müssen.

Die Klasse *CComboPicklist* benutzt zwei VFX-Systemtabellen: *Vfxpdef.dbf* und *Vfxplist.dbf*.

Die Tabelle *Vfxpdef.dbf* enthält die Beschreibungen der Auswahllisten. Für jede Auswahlliste gibt es einen Datensatz. Zu jeder Auswahlliste kann es Code geben, der ausgeführt wird, wenn der Benutzer eine Auswahl trifft. Dieser Code wird bei jeder Auswahl ausgeführt. In der Tabelle *Vfxplist.dbf* kann zu jedem Eintrag ein Code zugeordnet werden.

Die Tabelle *Vfxplist.dbf* enthält die auswählbaren Einträge. Das Feld *Picklist* enthält den Fremdschlüssel und zeigt auf einen korrespondierenden Datensatz in der Tabelle *Vfxpdef.dbf*. Die Felder *Code* und *Descript* enthalten Werte, die in der Auswahlliste angezeigt werden. Abhängig von der Einstellung der Auswahlliste in der Tabelle *Vfxpdef.dbf* kann nur die *Code*-Spalte oder die *Code*-Spalte und die *Descript*-Spalte angezeigt werden. Im Feld *Proccode* kann zu einem Eintrag Code eingetragen werden, der ausgeführt wird, wenn dieser Eintrag ausgewählt wird.

Für jede Verwendung der Klasse *CComboPicklist* kann eingestellt werden, ob neue Datensätze hinzugefügt werden dürfen, und welche Berechtigungsstufe Benutzer haben müssen, um neue Datensätze hinzufügen zu dürfen.

Eigenschaft:

nParentID – ID Schlüsselwert der Tabelle *Vfxpdef.dbf*.

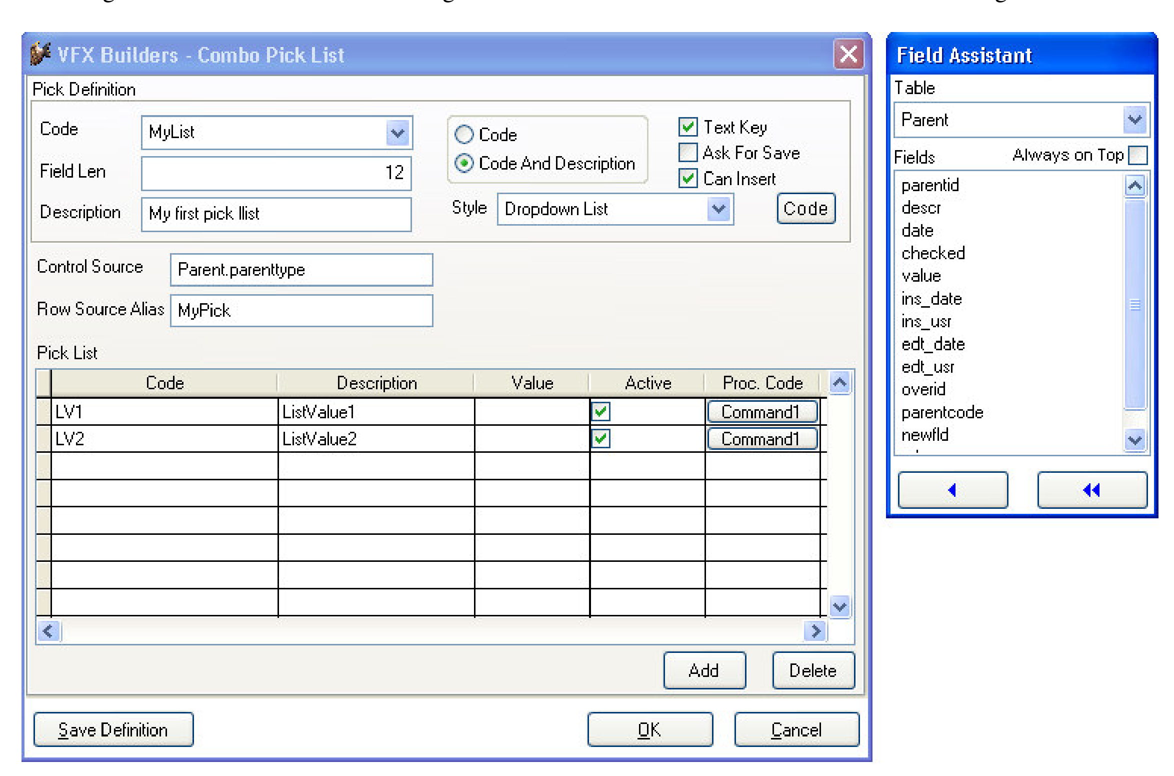
Methode:

Addnewcode – Diese Methode wird ausgeführt wenn der Benutzer einen neuen Wert in die Combobox einträgt. Wenn beim Hinzufügen von Werten weiterer Code ausgeführt werden soll, muss er in dieser Methode eingetragen werden.

Für die Klasse *CComboPicklist* können zwei Code-Blöcke in Tabellenfeldern hinterlegt werden. In der Tabelle *Vfxpdef.dbf* ist es das Memofeld *ProcCode* und in der Tabelle *Vfxplist.dbf* ist es das Memofeld *ProcCode*.

Der Code aus dem Feld *Vfxpdef.ProcCode* wird zur Laufzeit immer dann ausgeführt, wenn der Wert in der Combobox geändert wird. Der Code aus dem Feld *Vfxplist.ProcCode* ist einem bestimmten Eintrag zugeordnet und wird immer dann ausgeführt, wenn dieser Eintrag ausgewählt wird.

Für jeden Eintrag in der Tabelle *Vfxplist.dbf* kann eingestellt werden, ob es sich um einen aktiven Eintrag handelt. Durch dieses Verfahren brauchen Einträge, die zeitweise nicht zur Auswahl stehen sollen, nicht aus der Tabelle gelöscht werden. Um einen Eintrag zu deaktivieren muss der Wert im Feld *Active* auf .F. gesetzt werden.

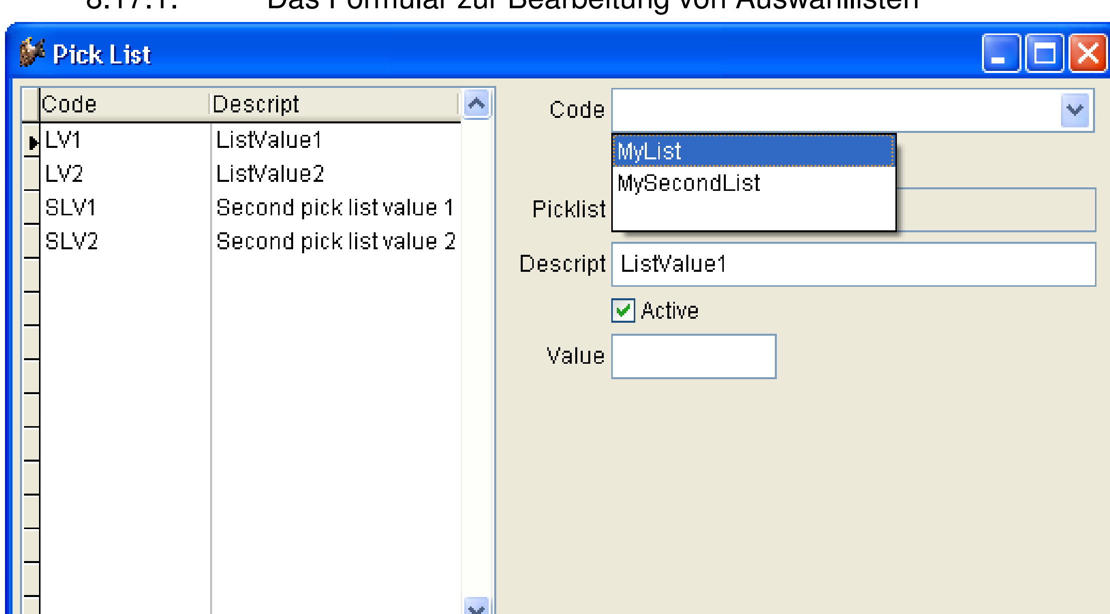


Die Klasse *CComboPicklist* sowie die Tabellen *Vfxpdef.dbf* und *Vfxplist.dbf* können mit dem VFX – Combo Pick List Builder bearbeitet werden.

Für die *CComboPicklist* müssen die Controlsource und der Alias für die *Rowsource* angegeben werden. Wenn der Alias für die *Rowsource* bereits in der Datenumgebung vorhanden ist, fragt der Builder, ob dieser Alias verwendet werden soll, oder ob eine weitere Instanz dieses Cursors der Datenumgebung hinzugefügt werden soll. Wenn der Alias für die *Rowsource* nicht in der Datenumgebung gefunden werden kann, wird das entsprechende

Cursor-Objekt vom Builder automatisch der Datenumgebung hinzugefügt und die Eigenschaften werden eingestellt.

8.17.1. Das Formular zur Bearbeitung von Auswahllisten



Dieses Formular kann Anwendern zur Bearbeitung von Auswahllisten zur Verfügung gestellt werden. Das Formular befindet sich in jedem VFX 11.0-Projekt und hat den Namen *VFXPlist.scx*.

Der Benutzer kann zwischen der Bearbeitung aller Datensätze wählen oder den sichtbaren Bereich durch einen Filter auf das Feld *Code* einschränken. Es ist möglich Datensätze zu löschen, aber Datensätze können auch als nicht aktiv markiert werden.

8.17.2. Die Klasse CComboPicklist

Diese Klasse dient zur einfachen Erstellung von Auswahllisten. Es können Auswahllisten erstellt werden, die nicht auf einer eigenen Tabelle basieren müssen.

Die Klasse *CComboPicklist* benutzt zwei VFX-Systemtabellen: *Vfxpdef.dbf* und *Vfxplist.dbf*.

Die Tabelle *Vfxpdef.dbf* enthält die Beschreibungen der Auswahllisten. Für jede Auswahlliste gibt es einen Datensatz. Zu jeder Auswahlliste kann es Code geben, der ausgeführt wird, wenn der Benutzer eine Auswahl trifft. Dieser Code wird bei jeder Auswahl ausgeführt. In der Tabelle *Vfxplist.dbf* kann zu jedem Eintrag ein Code zugeordnet werden.

Die Tabelle *Vfxplist.dbf* enthält die auswählbaren Einträge. Das Feld *Picklist* enthält den Fremdschlüssel und zeigt auf einen korrespondierenden Datensatz in der Tabelle *Vfxpdef.dbf*. Die Felder *Code* und *Descript* enthalten Werte, die in der Auswahlliste angezeigt werden. Abhängig von der Einstellung der Auswahlliste in der Tabelle *Vfxpdef.dbf* kann nur die *Code*-Spalte oder die *Code*-Spalte und die *Descript*-Spalte angezeigt werden. Im Feld *Proccode* kann zu einem Eintrag Code eingetragen werden, der ausgeführt wird, wenn dieser Eintrag ausgewählt wird.

Für jede Verwendung der Klasse *CComboPicklist* kann eingestellt werden, ob neue Datensätze hinzugefügt werden dürfen, und welche Berechtigungsstufe Benutzer haben müssen, um neue Datensätze hinzufügen zu dürfen.

Eigenschaft:

nParentID – ID Schlüsselwert der Tabelle *Vfxpdef.dbf*.

Methode:

Addnewcode – Diese Methode wird ausgeführt wenn der Benutzer einen neuen Wert in die Combobox einträgt. Wenn beim Hinzufügen von Werten weiterer Code ausgeführt werden soll, muss er in dieser Methode eingetragen werden.

Für die Klasse *CComboPicklist* können zwei Code-Blöcke in Tabellenfeldern hinterlegt werden. In der Tabelle *Vfxpdef.dbf* ist es das Memofeld *ProcCode* und in der Tabelle *Vfxplist.dbf* ist es das Memofeld *ProcCode*.

Der Code aus dem Feld *Vfxpdef.ProcCode* wird zur Laufzeit immer dann ausgeführt, wenn der Wert in der Combobox geändert wird. Der Code aus dem Feld *Vfxplist.ProcCode* ist einem bestimmten Eintrag zugeordnet und wird immer dann ausgeführt, wenn dieser Eintrag ausgewählt wird.

Für jeden Eintrag in der Tabelle *Vfxplist.dbf* kann eingestellt werden, ob es sich um einen aktiven Eintrag handelt. Durch dieses Verfahren brauchen Einträge, die zeitweise nicht zur Auswahl stehen sollen, nicht aus der Tabelle gelöscht werden. Um einen Eintrag zu deaktivieren muss der Wert im Feld *Active* auf .F. gesetzt werden.

8.18. VFX – Parent/Child Builder

Obwohl es einen speziellen VFX-Builder zur Erstellung von 1:n-Formularen gibt, ist es manchmal besser, Child-Daten in einem eigenen Formular zu bearbeiten. Das ist insbesondere dann der Fall, wenn Sie das Child-Formular auch für die direkte Bearbeitung einsetzen und nicht nur durch das Hauptformular einsetzen wollen. Wenn Sie außerdem viele Felder auf dem Child-Formular haben, kann es schwierig werden, diese in einem 1:n-Formular zu bearbeiten.

Eine besondere Stärke von VFX ist die Verwendung der Linked Child-Technik. Dabei werden zwei Formulare logisch miteinander verbunden. Ein Formular dient dabei als Parent-Formular. Als Parent-Formular kann jede VFX-Formularklasse dienen. Auch das Child-Formular kann auf jeder VFX-Formularklasse basieren.

Beim Bewegen des Satzzeigers im Parent-Formular wird die Ansicht im Child-Formular automatisch aktualisiert und es werden die zum aktuellen Parent gehörenden Datensätze angezeigt.

Wenn das Child-Formular auf einer Tabelle basiert, wird ein Filter verwendet, um den sichtbaren Datenbereich einzuschränken. Wenn das Child-Formular auf einer Ansicht basiert, wird bei Bedarf ein REQUERY() durchgeführt um die gewünschte Datenmenge anzuzeigen. Die zugrunde liegende Ansicht darf dabei genau einen variablen Ansichtsparameter haben, der dem Parent-Schlüssel entsprechen muss.

Ein Parent-Formular kann mehrere, verschiedene Child-Formulare aufrufen. Ein Child-Formular kann wiederum als Parent für andere Child-Formulare dienen.

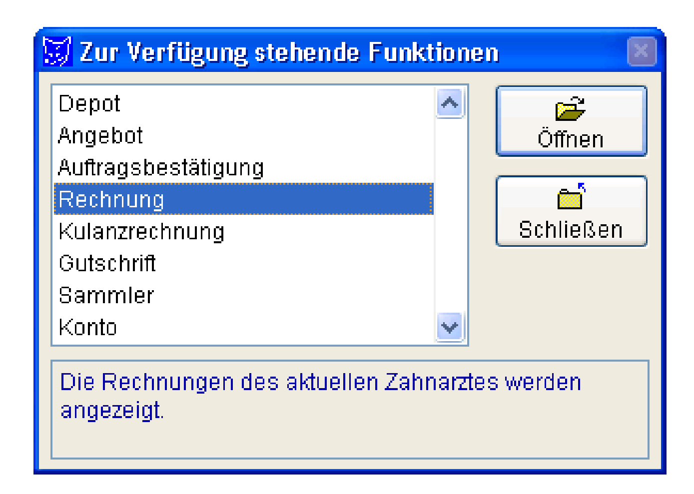
8.18.1. Vorbereitung des Parent-Formulars

Beim Parent-Formular müssen mit dem Form Builder die Optionen *Has More Options* (setzt die Eigenschaft *lmore* auf .T.), *Has Child Form* und *Auto Sync Child Form* (setzt die Eigenschaft *lautosynchildform* auf .T.) ausgewählt werden. Der Form Builder trägt automatisch Template-Code in die Methoden *OnMore()* und *onsetchilddata()* ein. Mithilfe der Methode *OnMore()* wird das Child-Formular aufgerufen.

Wenn der Benutzer die verfügbaren Optionen zum aktuellen Parent-Datensatz sehen will, hat er verschiedene Möglichkeiten:

- Er kann die Funktionstaste *F6* drücken.
- Er wählt *Weitere Funktionen...* im *Bearbeiten*-Menü.
- Er drückt auf die *Weitere Funktionen*-Schaltfläche in der Standard-Symbolleiste.

Abhängig von dem Code in der Methode *OnMore()* wird der Benutzer einen Dialog sehen, der so ähnlich aussieht wie der folgende:



Der Aufruf der *OnMore()*-Methode mit dem Parameter *tnPassThrough* ist sehr nützlich, wenn Sie ein Formular direkt über die zugeordnete Zahl starten wollen. Sie können diese Technik benutzen, um ein Formular aus der *OnMore()*-Methode über eine Schaltfläche aus einer Symbolleiste zu starten.

Wenn es nur eine Option in der *OnMore()*-Methode gibt, wird das zugeordnete Formular geöffnet, ohne dass dieser Dialog erscheint.

8.18.2. Vorbereiten des Child-Formulars

Der VFX-Entwickler muss im Child-Formular mit dem Form Builder auf der Seite Optionen *Is Child Form* auswählen oder manuell die Formulareigenschaft *lchildform* auf .T. zu setzen.

Wenn Sie ein Formular aufrufen, übergeben Sie die benötigten Parameter an das *Init()*-Ereignis dieses Formulars. Da die übergebenen Parameter nicht automatisch für andere Methoden des gleichen Formulars sichtbar sind, speichern VFX-Formulare die benötigten Parameter in speziellen Eigenschaften.

Hier ist der Code des *Init()*-Ereignis, den der VFX-Formular-Builder als Vorlage für Ihre Bedürfnisse erzeugt:

```
lparameters tcArg

local lInitOk

if !empty(tcArg)
        if getArgCount(tcArg) <> 0

                this.cCalledBy       = upper(  getArg(tcArg,1)         )
                this.cFixFieldValue = strtran(getArg(tcArg,2),"@",";")
                this.Caption         =         getArg(tcArg,3)
                this.cFixFieldName   = strtran(getArg(tcArg,4),"@",";")
                this.cFilterExpr     = upper(  getArg(tcArg,5)         )

                this.lPutInLastFile  = .f.

                ***********************************************************************
                ** Set who has called you

                if this.cCalledBy = "<CalledBy>"

        *******************************************************************
                        ** Disable CPickField that are Fix Fields for this form

                        *{PickFieldList}*
                endif
        endif
endif

this.SetQueryArg()

lInitOk =dodefault(tcArg)

********************************************************
** Insert your extra initialization code here
```

```
return lInitOk
```

Der Vorlagencode kann so aussehen, wenn Sie ihn an Ihre Bedürfnisse angepasst haben:

```
lparameters tcArg

local lInitOk

if !empty(tcArg)

	if getArgCount(tcArg) <> 0
		this.cCalledBy      = upper(  getArg(tcArg,1)        )
		this.cFixFieldValue = strtran(getArg(tcArg,2),"@",";")
		this.Caption        =         getArg(tcArg,3)
		this.cFixFieldName  = strtran(getArg(tcArg,4),"@",";")
		this.cFilterExpr    = upper(  getArg(tcArg,5)        )

		this.lPutInLastFile  = .f.
		***********************************************************************
		** Set who has called you

		if this.cCalledBy = "PARENT"
		*******************************************************************
			** Disable CPickField that are Fix Fields for this form

			ThisForm.pgfPageFrame.Page1.cntParentid.lFixField = .t.

		endif
	endif
endif

this.SetQueryArg()

lInitOk =dodefault(tcArg)

********************************************************
** Insert your extra initialization code here

return lInitOk
```

Die VFX-Funktion *getArg()* überprüft die Parameterzeichenkette und zerlegt sie in ihre Bestandteile. Die Bestandteile sind durch Semikolon getrennt.

Sehen Sie sich das Beispiel an. Der übergebene Parameter kann die folgende Zusammensetzung haben, wenn wir das Kontakt-Formular zu einer bestimmten Firma aufrufen:

```
"COMP;1234567890;Kontakte zur Firma ISYS;CONT_COMP_ID;UPPER(CONT_COMP_ID)= '1234567890'"
```

Die individuellen Teile dieser Zeichenkette werden in den bereitgestellten Formulareigenschaften gespeichert, bevor sie innerhalb des ganzen Formulars benutzt werden können. Lassen Sie uns die Formulareigenschaften anschauen, die die Informationen aus der übergebenen Parameterzeichenkette *tcArg* speichern:

VFX-Formulareigenschaft	***Beschreibung***	***Beispiel***
cCalledBy	Diese Zeichenkette gibt an, aus welchem Formular dieses Formular aufgerufen wurde.	COMP
cFixFieldValue	Der Wert des Feldes aus der Haupttabelle (**Primärschlüssel in der Haupttabelle**).	1234567890
Caption	Titel des Child-Formulars. Hier ist ein Hinweis auf den zugehörigen Parent-Datensatz sinnvoll.	Kontakte zur Firma ISYS
cFixFieldName	Der Name des Feldes in der Child-Tabelle, der die 1:n-Beziehung definiert. Dieses Feld erhält den oben angegebenen Wert, wenn ein neuer Datensatz hinzugefügt wird (**Fremdschlüssel in der Child-Tabelle**).	CONT_COMP_ID
cFilterExpr	Der (idealerweise) Rushmore-optimierte Filterausdruck, um die Datensätze entsprechend dem Kriterium der Haupttabelle anzuzeigen.	UPPER(CONT_COMP_ID) = '1234567890'

Bei sehr großen Datenmengen kann es besser sein, mit Ansichten zu arbeiten. Die VFX-Mechanismen arbeiten grundsätzlich genauso. Wenn die Child-Daten aus einer Ansicht stammen, brauchen Sie den Filterausdruck nicht zu übergeben.

8.18.3. Einstellungen im VFX – Parent/Child Builder

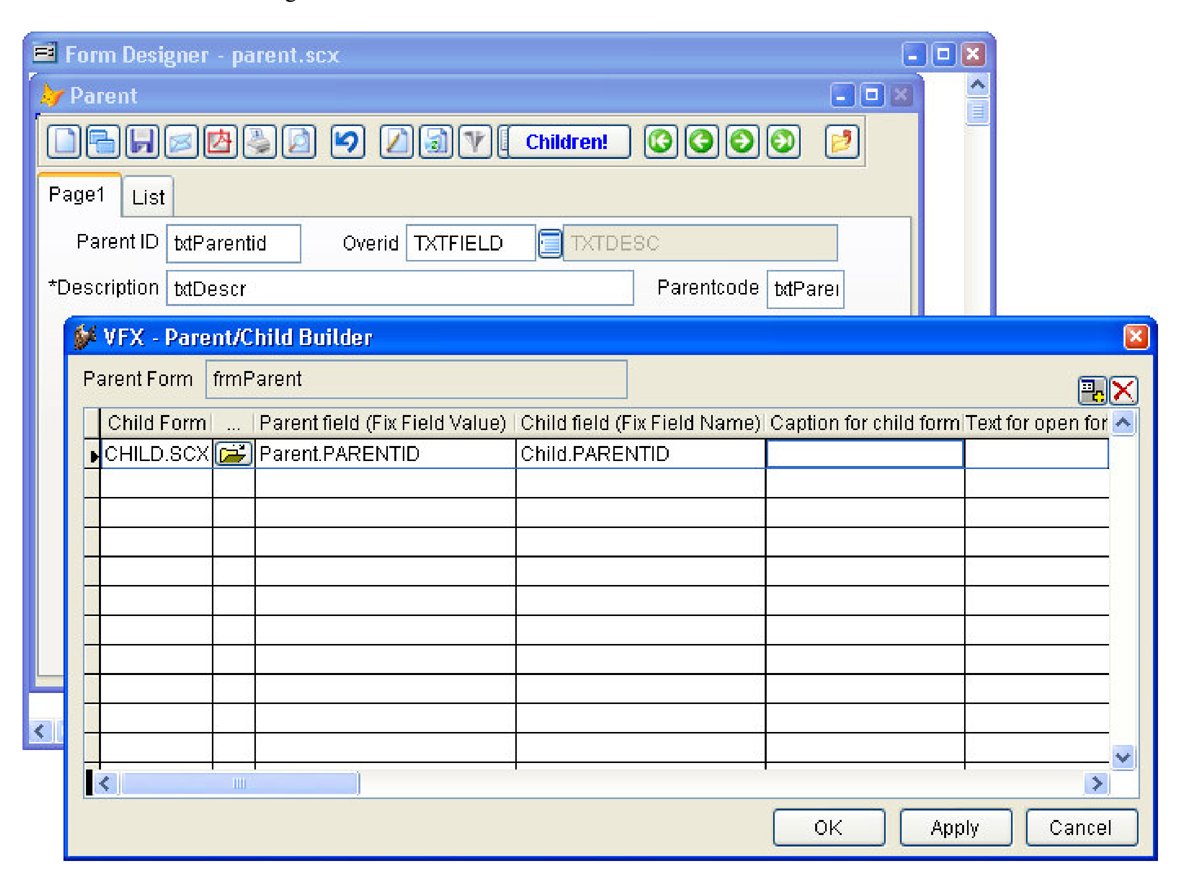
Durch Einstellen von wenigen Eigenschaften in der *OnMore()*-Methode eines Parent-Formulars kann ein Child-Formular gestartet werden. Dem Child-Formular wird der Schlüssel des Parent-Formulars übergeben. Im Child-Formular sind nur die Daten sichtbar, die dem Schlüssel des Parent-Datensatzes entsprechen. Der im Child-Formular sichtbare Bereich kann wahlweise mit einem Filter oder einer Ansicht eingeschränkt werden.

Durch Einstellen einiger Eigenschaften in der *OnSetChildData()*-Methode des Parent-Formulars wird aus dem einfachen Child-Formular ein Linked Child-Formular. Das heißt, wenn im Parent-Formular der Satzzeiger bewegt wird, wird automatisch die Ansicht im Child-Formular entsprechend dem Parent-Schlüssel aktualisiert.

Es ist möglich von einem Parent-Formular mehrere Linked Child-Formulare gleichzeitig zu steuern. Als Formulartyp kommen sowohl für das Parent-Formular als auch für das Child-Formular alle VFX-Formulartypen in Frage. Es ist möglich eine 1:n:m-Beziehung zu realisieren, indem als Linked Child ein OneToMany-Formular verwendet wird.

In VFX 11.0 gibt es einen Builder zur Bearbeitung von Parent/Child-Beziehungen. Zur einfacheren Verwaltung von Parent/Child-Beziehungen gibt es die neue Klasse *CChildManager*. Zur Verwendung des VFX – Parent/Child Builder muss zunächst das Parent-Formular im VFP Formular-Designer geöffnet werden. Dann kann der VFX – Parent/Child Builder aus dem VFX 11.0 Menü gestartet werden.

Im Builder können beliebig viele Child-Formulare verwaltet werden.

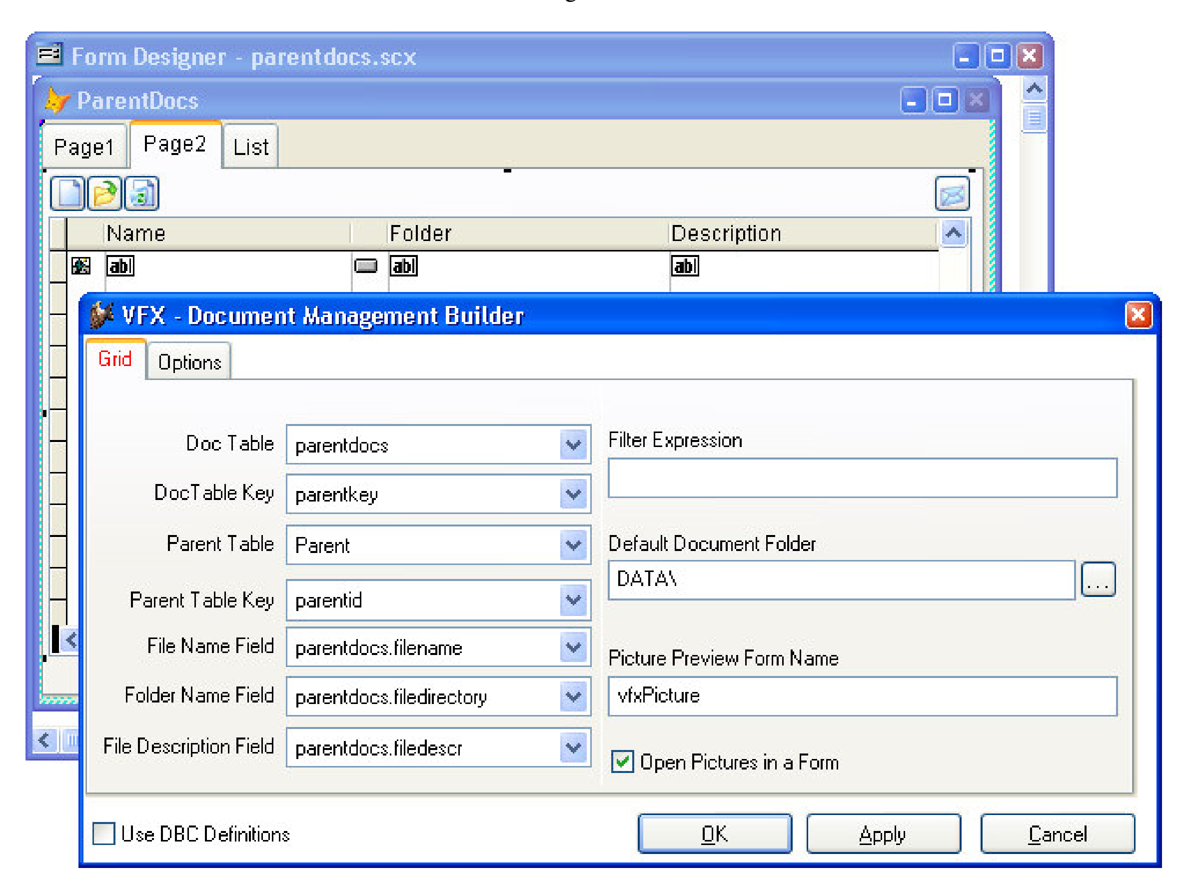


In der Spalte *Child Form* kann der Name eines Child-Formulars über die Öffnen-Schaltfläche ausgewählt werden. In der Spalte *Parent field (Fix Field Value)* wird der Name des ID-Feldes der Parent-Tabelle eingetragen. Der Wert dieses Feldes wird an das Child-Formular beim Start und bei jeder Bewegung des Satzzeigers im Parent-Formular übergeben.

In der Spalte *Child field (Fix Field Name)* wird der dazugehörige Fremdschlüssel aus der Child-Tabelle eingetragen.

8.19. VFX – Document Management Builder

Die neue Klasse *CDocumentManagement* dient zur Verwaltung von Dokumenten aller Art (z. B. Word, Excel, Powerpoint) innerhalb einer Anwendung. Die Klasse *CDocumentManagement* ist ein Container, der Child-Datensätze zum aktuellen Datensatz im Formular verwaltet. Die Dokumentenverwaltung ermöglicht dem Anwender Dokumente zu öffnen und als E-Mailanhang zu versenden.



Diese Klasse kann bestehenden Formularen einfach hinzugefügt werden.

cDefaultDocumentFolder – Standardordner für Dokumente.

cFilterExpression – Anzuwendender Filterausdruck.

lOpenPicturesInForm – Wenn der Wert dieser Eigenschaft auf .T. eingestellt ist, werden Bilddateien in einem VFX-Formular angezeigt. Der Name des Formulars kann in der Eigenschaft *cPicturePreviewFormname* angegeben werden. Wenn der Wert dieser Eigenschaft auf .F. eingestellt ist, werden Bilddateien mit der Anwendung geöffnet, die im Windows-Explorer mit der Namenserweiterung der Datei verknüpft ist. Der Standardwert ist .F.

cPicturePreviewFormname – Name des Formulars zur Vorschau auf Bilddateien. Der Standardwert ist *VFXPicture*.

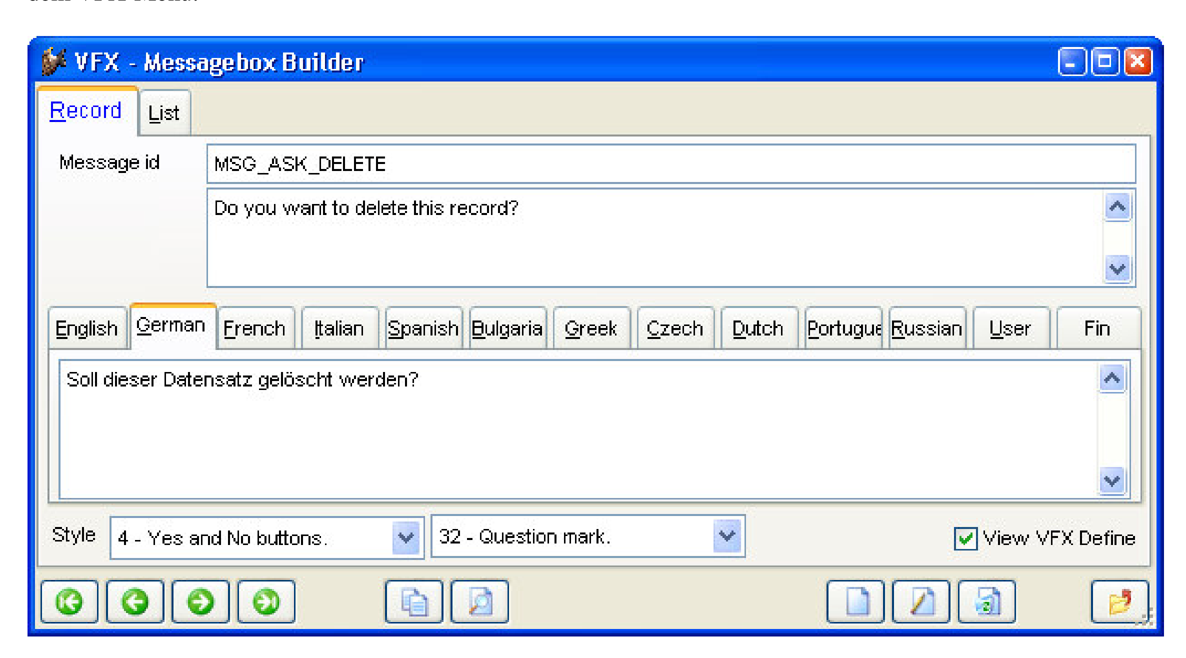
cPicturePreviewCaption – Der hier zugewiesene Text wird dem Formular zur Vorschau auf Bilddateien als Caption mitgegeben.

8.20. VFX – Messagebox Builder

Ein nützliches Werkzeug zur Erstellung von Messageboxen in verschiedenen Sprachen ist der VFX – Messagebox Builder. Die Texte der Messagebox werden in der Tabelle *Vfxmsg.dbf* gespeichert. Der Befehl zur Anzeige der Messagebox wird in die Zwischenablage kopiert und kann von dort in den eigenen Programm-quelltext übernommen werden. Dabei wird nicht der Text selbst, sondern eine Konstante als Parameter über-

geben. Die Include-Dateien mit den Werten der Konstanten in der gewünschten Sprache werden mit dem VFX – Message Editor erstellt.

Um den VFX – Messagebox Builder aufzurufen, wählen Sie den Menüpunkt *Form, MessageBox Builder* aus dem VFX-Menü.



Klicken Sie auf die Schaltfläche *neu* um eine neue Messagebox anzulegen. Tragen Sie dann im Feld *Message id* eine eindeutige Bezeichnung für die Messagebox ein. Im Seitenrahmen können Sie für jede benötigte Sprache den Text hinterlegen.

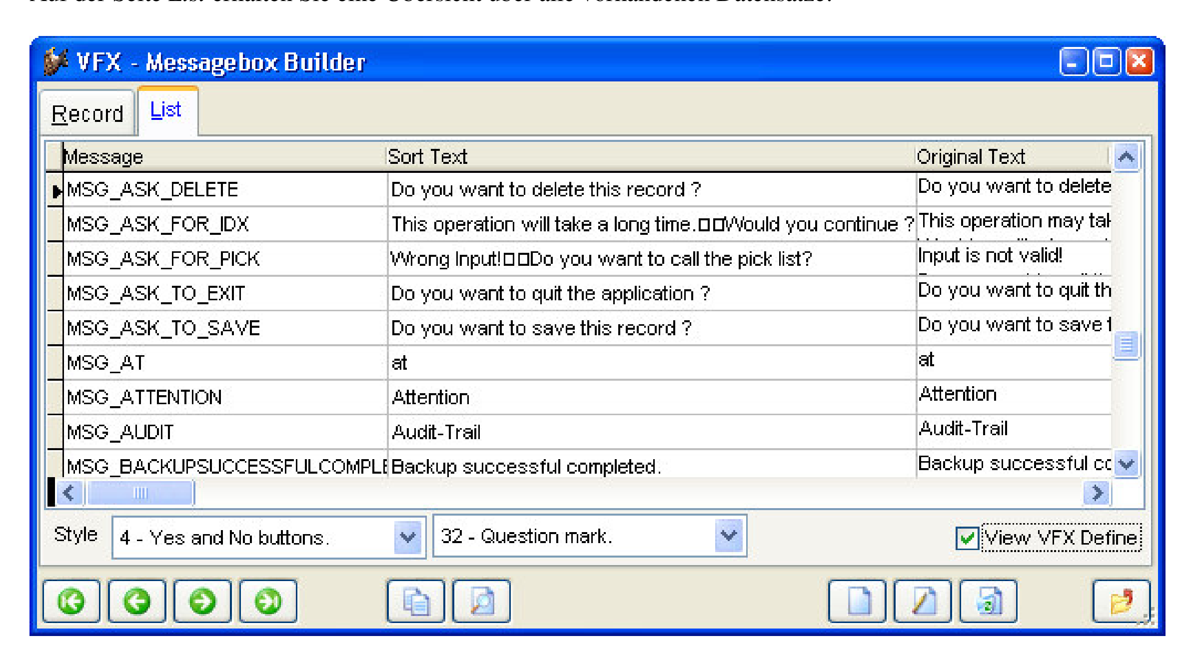
In der Zeile *Style* wählen Sie gewünschten Typ der Messagebox aus. Es kann zwischen verschiedenen Symbolen und Schaltflächen auf der Messagebox ausgewählt werden.

Durch einen Klick auf die Schaltfläche *Test it!* wird die Messagebox in der Vorschau angezeigt.

Kopieren Sie den vom VFX – Messagebox Builder erstellten Code mit der Schaltfläche *Copy code to clipboard* in die Zwischenablage. Aus der Zwischenablage kann der Code in einem beliebigen Programmteil eingefügt werden.

Der VFX – Messagebox Builder legt für jeden Eintrag einen Datensatz in der Tabelle *Vfxmsg.dbf* an.

Auf der Seite *List* erhalten Sie eine Übersicht über alle vorhandenen Datensätze:



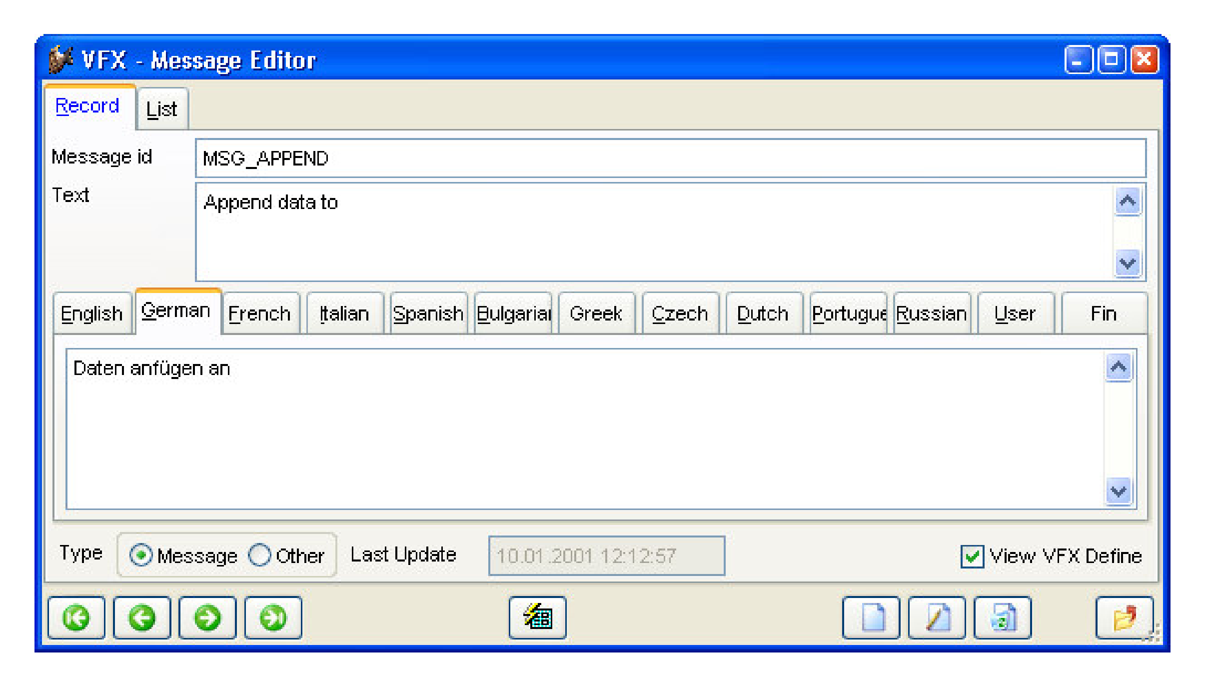
Tipp: Auch wenn Sie keine mehrsprachigen Anwendungen erstellen, können Sie den VFX – Messagebox Builder einsetzen.

8.21. VFX – Message Editor

Die Werte aller von VFX verwendeten Konstanten stehen in der freien Tabelle *Vfxmsg.dbf*. Für jede Sprache ist ein Memofeld mit dem Text vorhanden. Mit dem VFX – Message Editor können diese Texte bearbeitet werden.

Der VFX – Message Editor ist der Zentrale Ort um, alle Bezeichnungen, Meldungen, Tooltip-Texte und Statuszeilenmeldungen zu verwalten und in andere Sprachen zu übersetzen. Aus dem VFX – Message Editor heraus können Sie die benötigten Include-Dateien (*Usertxt.h* und *Usermsg.h*) erstellen.

Um den VFX – Message Editor aufzurufen, wählen Sie den Menüpunkt *Form, Message Editor* aus dem VFX-Menü.



Klicken Sie auf die Schaltfläche *Make Include File* um eine Include-Datei in der im Seitenrahmen angezeigten Sprache zu erstellen. Die Include-Dateien werden in einem Ordner mit der Bezeichnung der jeweiligen Sprache unterhalb des Include-Ordners Ihres Projektes gespeichert.

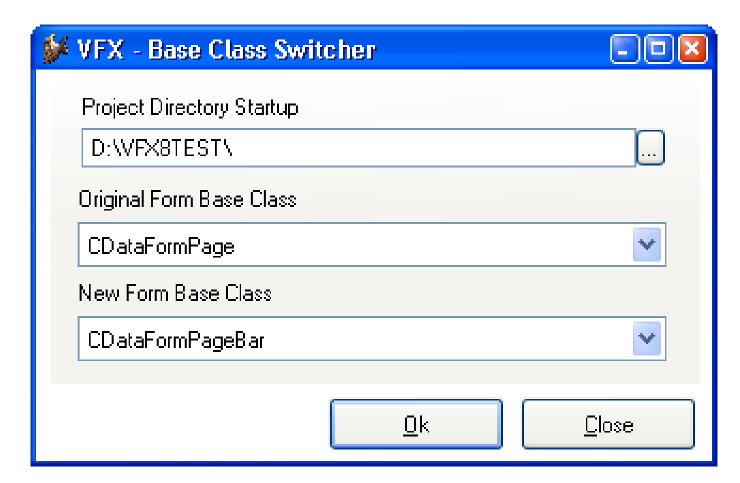
Nach der Erstellung Ihrer Include-Dateien müssen Sie diese nur noch in den *\INCLUDE*-Ordner Ihres Projektes kopieren, wie im Kapitel *Erstellen mehrsprachiger Anwendungen* beschrieben ist.

Tipp: Sie können Ihre eigenen Konstanten mit den erzeugten Konstanten aus der Tabelle *Vfxmsg.dbf* mischen. Schreiben Sie Ihre Konstanten vor oder nach dem VFX-Header bzw. -Footer.

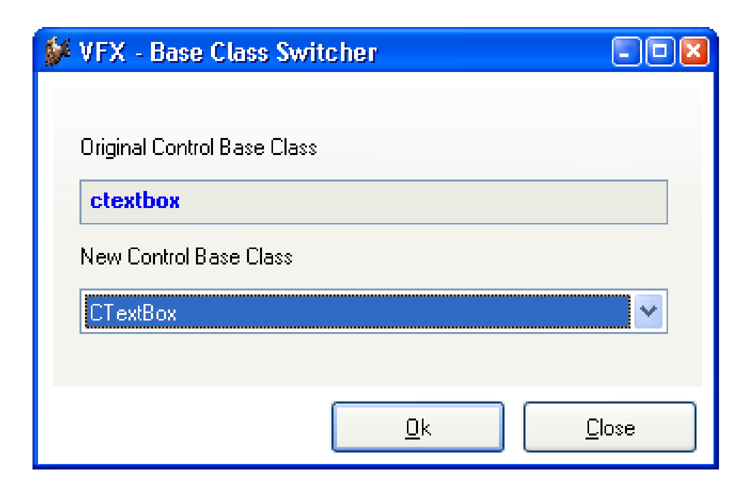
8.22. VFX – Class Switcher

Der Class Switcher hat zwei Funktionen.

Wenn beim Aufruf kein Formular geöffnet ist, ändert der Class Switcher die Klassen von Formularen in einem ganzen Projekt. Zum Beispiel kann die Formularklasse *CDataFormPageBar* durch *CDataFormPage* ersetzt werden. Dadurch ist es möglich alle Formulare mit Schaltflächen auszustatten bzw. diese wieder zu entfernen. Als besonders hilfreich erweist sich dieses Werkzeug bei der Aktualisierung vorhandener VFX 3-Projekte. In VFX 3 hatte jedes Formular am unteren Rand eine Leiste mit Schaltflächen. In VFX 11.0 kann man stattdessen eine richtige Symbolleiste verwenden.

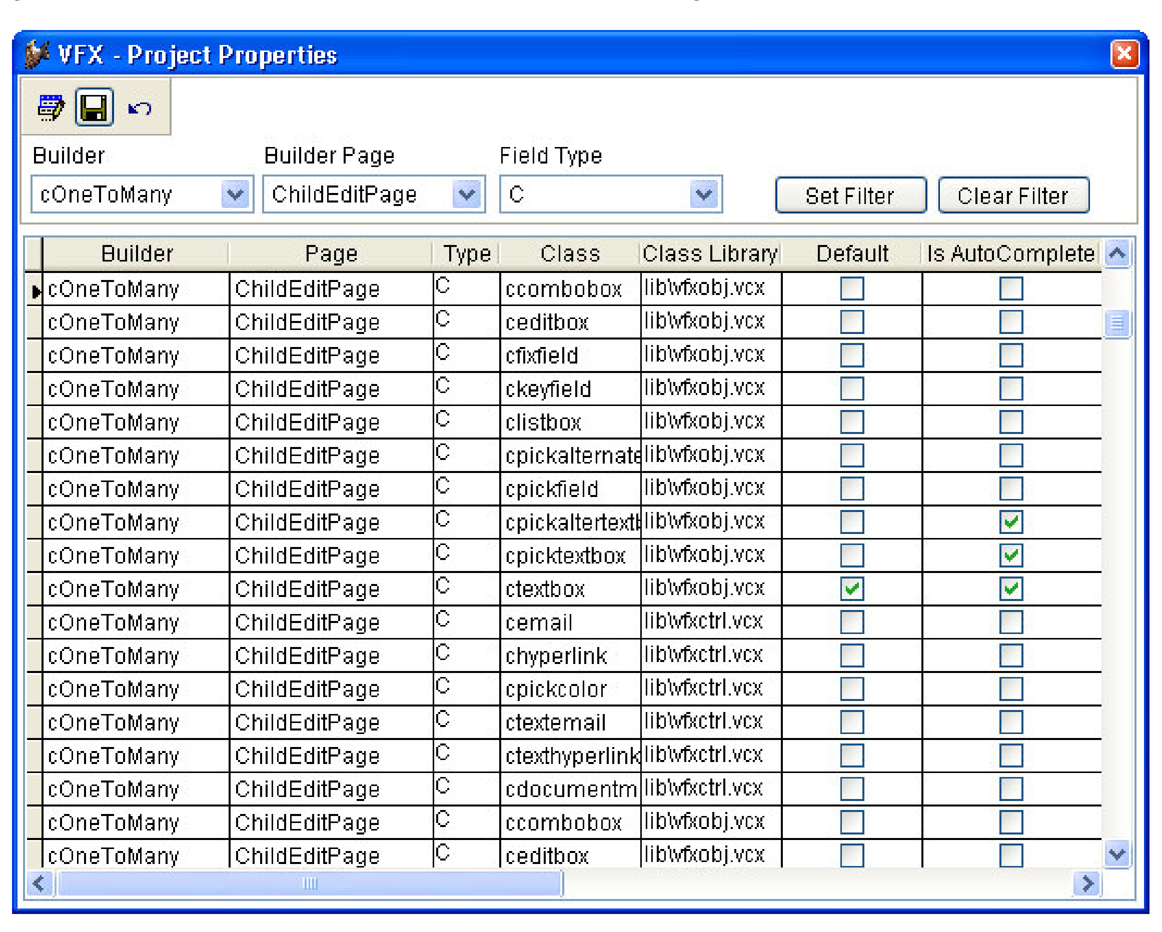


Wenn beim Aufruf des VFX – Class Switcher ein Formular zur Bearbeitung geöffnet ist, können die einzelnen Objekten zugrunde liegenden Klassen geändert werden. So ist es z. B. möglich, aus einer Textbox nachträglich ein Drehfeld zu machen.



8.23. VFX – Project Properties

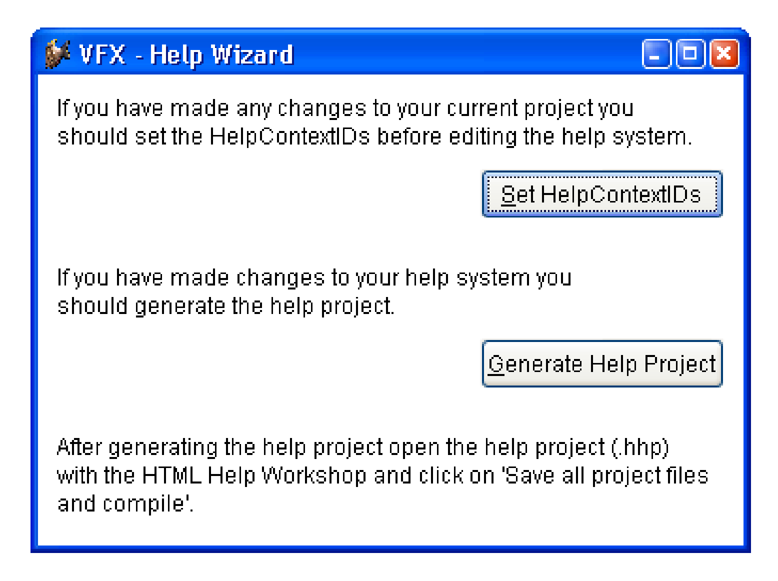
In VFX können eigene Ableitungen der VFX-Klassen verwendet werden. Im Dialog VFX – Project Properties können die zu verwendenden Klassen für die einzelnen Steuerelement-Typen eingetragen werden. Als Vorgabe stehen hier die bekannten Klassen aus der Klassenbibliothek *Vfxobj.vcx*. Der VFX-Entwickler kann diese Vorgaben ändern und eigene Klassen, die vorzugsweise in der Klassenbibliothek *Appl.vcx* gespeichert sind, eintragen. Diese Klassen können von den VFX-Buildern bei der Erstellung neuer Formulare verwendet werden.



8.24. VFX – Help Wizard

In VFX ist ein System zur Erstellung von CHM-Hilfedateien integriert.

Der VFX – Help Wizard trägt in alle Steuerelemente eines Projekts automatisch eindeutige *HelpContextIDs* ein.



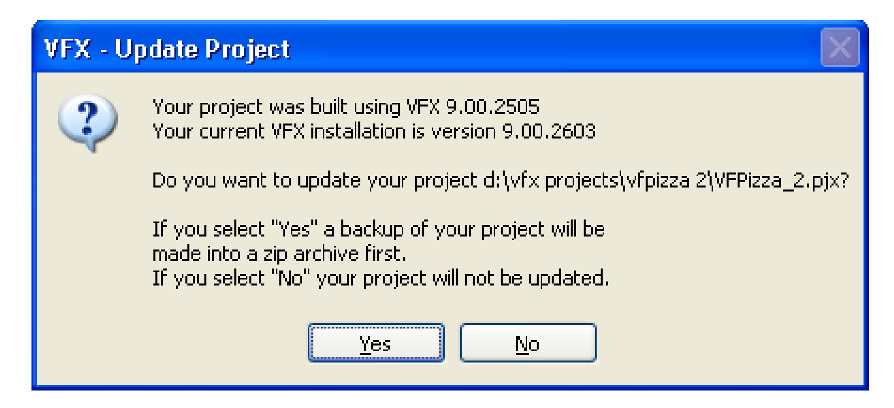
Wenn zur Laufzeit der Anwendung die Tabelle *Vfxhelp.dbf* zur Verfügung steht, können Hilfetexte in diese Tabelle erfasst werden. Dafür wird das Formular *Vfxhelp.scx* geöffnet. Der eigentliche Hilfetext wird in einer Editbox erfasst und in der Tabelle *Vfxhelp.dbf* gespeichert.

Mittels des VFX – Help Wizard können aus den Daten der Tabelle *Vfxhelp.dbf* vollautomatisch HTM-Dateien sowie ein Hilfe-Projekt erstellt werden. Mit dem Help-Workshop muss dieses Projekt nur noch kompiliert werden und die CHM-Hilfedatei mit kontextsensitiver Hilfe zur gesamten Anwendung ist fertig.

Wenn die Tabelle *Vfxhelp.dbf* zur Laufzeit der Anwendung nicht zur Verfügung steht, wird das normale kontextsensitive Hilfesystem aktiviert. Die CHM-Hilfedatei wird geöffnet und als Parameter wird die *HelpContextID* des aktiven Steuerelements übergeben.

8.25. VFX – Project Update Wizard

Projekte, die mit älteren Versionen von VFX oder mit älteren Builds von VFX 11.0 erstellt wurden, können jetzt automatisch auf die neueste Version aktualisiert werden.



Der VFX – Project Update Wizard kann direkt aus dem VFX 11.0-Menü über den Menüpunkt *Project, Update Project* gestartet werden. Der VFX – Project Update Wizard vergleicht die Version des geöffneten Projekts mit der installierten VFX 11.0-Version. Wenn das Projekt mit einer älteren VFX-Version erstellt wurde, wird der Entwickler gefragt, ob das Projekt aktualisiert werden soll.

Nach einem Klick auf Ja beginnt der Wizard mit der Arbeit. Zunächst wird zur Sicherheit eine Sicherungskopie des Projekts in einer Zip-Datei angelegt. Die Zip-Datei wird im Projektordner angelegt und erhält den Namen der Projektdatei. Wenn das Archiv bereits existiert oder nicht angelegt werden kann, beendet der Wizard seine Arbeit sofort.

Der VFX – Project Update Wizard aktualisiert die VFX-Klassenbibliotheken, VFX-Berichtsvorlagen und die Datei *Vfxfunc.prg*. Der Tabelle *Vfxmsg.dbf* werden gegebenenfalls neu hinzugekommene Datensätze hinzugefügt. Alle Include-Dateien werden neu erstellt.

Die Struktur der freien VFX-Tabellen wird aktualisiert. Fehlende Felder oder Indexschlüssel werden automatisch ergänzt.

Fehlende Dateien werden dem Projekt automatisch hinzugefügt, wie zum Beispiel neue Bitmap-Dateien oder freie Tabellen.

Damit hat der VFX Update Project Wizard seine Aufgabe getan und hat uns damit viel Arbeit abgenommen. In der Regel werden die so aktualisierten Projekte sofort mit der neuen VFX 11.0-Version lauffähig sein. Trotzdem sollte der Entwickler das Projekt sorgfältig prüfen und bei Bedarf manuelle Ergänzungen machen.

Die meisten Anwendungen werden zum Beispiel ein speziell angepasstes Menü *Vfxmenu* haben. Der Update Projekt Wizard kann nicht wissen, welche Menüeinträge der Entwickler vielleicht absichtlich entfernt hat. Der Wizard kann daher keine neuen Menüeinträge hinzufügen. Durch einen Vergleich mit dem Menü aus der VFX 11.0-Installation können Menüeinträge für neue Funktionen aber schnell ergänzt werden.

Prüfen Sie das neue *Vfxmain.prg* und machen Sie von Hand die für Ihr Projekt erforderlichen Änderungen.

In bisherigen Versionen von VFX wurden public Variablen für die Felder aus den Datensätzen der Tabellen *Vfxsys.dbf* und *Vfxuser.dbf* angelegt. In VFX 11.0 werden stattdessen Eigenschaften von Objekten verwendet.

Beispiel:

```
Alt: gu_meinFeld  Neu: goUser.meinFeld
Alt: gs_meinFeld  Neu: goSystem.meinFeld
```

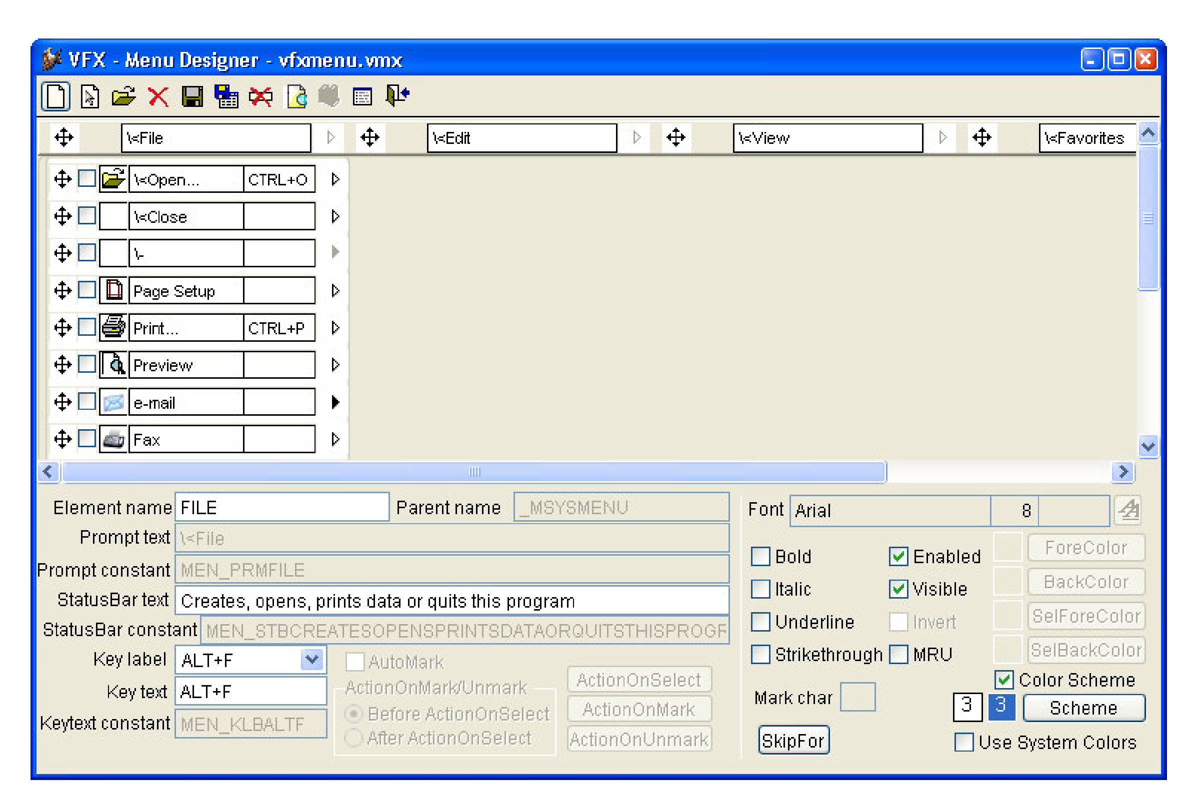
Sie können alle Verweise auf *gu_* bzw. *gs_* in Ihren Projekten mit dem Code Reference Tool aus VFP 9 finden und so alle betreffenden Code-Stellen einfach und schnell ändern.

8.26. PDM – Project Documenting

Eine speziell für VFX entwickelte Version des Projekt- und Datenbank-Dokumentations-Tools PDM wird mit VFX geliefert. Das PDM kann über den VFX 11.0-Menüpunkt *Project, Project Documenting* gestartet werden und fertigt zu einem Projekt vollautomatisch eine vollständige technische Dokumentation an. Die Dokumentation wird im HTML-Format erstellt und enthält zahlreiche Querverweise.

8.27. VFX Menü-Designer

Der VFX Menü-Designer (VMD) ist ein Werkzeug zur schnellen Entwicklung von Menüs. Der VMD ist ein visueller Designer, in dem das Menü schon während der Entwicklung so angezeigt wird, wie es zur Laufzeit aussehen wird. Der VMD macht die Entwicklung einfacher und ermöglicht die schnelle Einstellung aller Menü-Eigenschaften im Gegensatz vom VFP Menü-Designer, der nicht alle Eigenschaften von Menüs unterstützt. Es können mehrsprachige Menüs erstellt werden, indem auf die entsprechende Schaltfläche in der Symbolleiste geklickt wird.



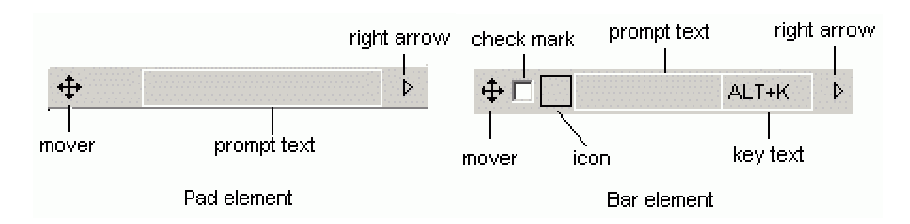
Ein in einem VFX-Projekt enthaltenes Menü kann direkt aus dem VFP-Projekt-Manager mit dem VMD geöffnet werden. Wahlweise können Menüs auch aus dem VMD heraus über das Öffnen-Symbol in der Symbolleiste oder über den entsprechenden Menüpunkt geöffnet werden. Im Öffnen-Dialog kann zwischen den Menütypen *.mnx* und *.vmx* gewechselt werden. Wenn ein Menü geöffnet wird, das noch nie mit dem VMD bearbeitet wurde, wird es automatisch in das *.vmx*-Format konvertiert.

Das geöffnete Menü kann visuell bearbeitet werden. Es können Einträge hinzugefügt und gelöscht werden und es können die Eigenschaften der einzelnen Einträge bearbeitet werden.

Neue Menü-Pads können durch einen Klick auf den Rechtspfeil, der sich rechts neben jedem Menüeintrag befindet, angelegt werden. Dadurch wird ein Untermenü angelegt. Einem Menü können neue Einträge hinzugefügt werden, indem auf den Pfeil nach unten unterhalb des Eintrags geklickt wird.

Ein Menüeintrag oder ein Menü-Pad können gelöscht werden, wenn sich der Fokus darauf befindet. Über den Menüpunkt löschen oder mit der Tastenkombination *Strg+Entf* wird der markierte Eintrag gelöscht.

Einige der Eigenschaften eines Menüeintrags können visuell eingestellt werden:



- Prompt text

Der angezeigte Text kann direkt eingetragen werden, wenn sich der Fokus auf dem jeweiligen Eintrag befindet. Die Textbox *Prompt text* im unteren Teil des VMD dient nur zur Anzeige des aktiven Eintrags im visuellen Teil des Designers.

- Key text

Die Bezeichnung des Tastenschlüssels zeigt dem Anwender die Zugriffstaste oder Tastenkombination an, mit der der Eintrag ausgewählt werden kann. Die Bezeichnung sollte dem im unteren Teil des VMD gewählten Tastenschlüssels entsprechen.

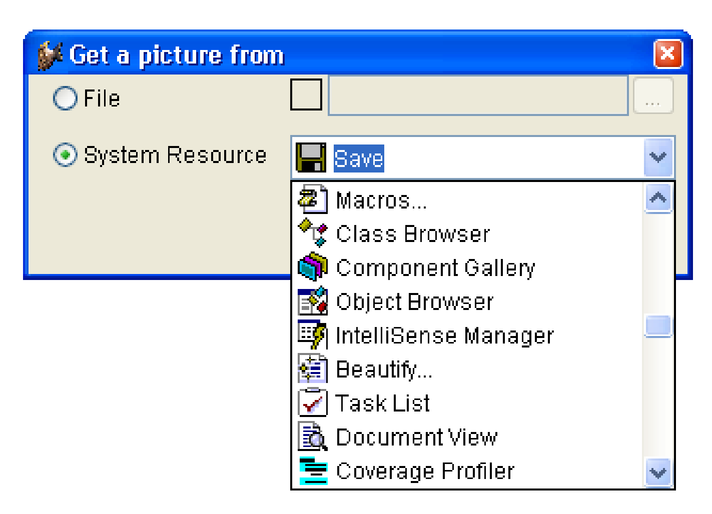
- Check mark

Damit sich ein Menüeintrag wie ein Kontrollkästchen verhält, muss bei *AutoMark* eine Markierung gesetzt werden. Wenn zusätzlich eine Markierung bei *Check mark* gesetzt wird, ist der Menüeintrag bereits beim Laden des Menüs markiert. Für Menüeinträge, die sich wie ein Kontrollkästchen verhalten, können zusätzliche Code-Teile ausgeführt werden, wenn das entsprechende Kontrollkästchen markiert wird (*ActionOnMark*) bzw. wenn die Markierung aufgehoben wird (*ActionOnUnmark*). Im Standard-VFP-Editorfenster kann der jeweilige Code bearbeitet werden.

Der Code, der bei *ActionOnMark* oder *ActionOnUnmark* eingegeben wird, kann wahlweise vor oder nach der *ActionOnSelect* ausgeführt werden. Um dieses Verhalten einzustellen, ist die entsprechende Option „*Before ActionOnSelect*“ oder „*After ActionOnSelect*“ auszuwählen.

- Icon

Jedem Eintrag in einem Menü kann ein Symbol zugeordnet werden. Dieses Symbol kann aus den in VFP integrierten Systemressourcen ausgewählt werden oder es kann eine Datei verwendet werden. Durch einen Klick auf das schwarzumrandete Kästchen kann ein Symbol mithilfe des *Get a picture from*-Dialogs ausgewählt werden. In diesem Dialog kann zwischen einer Datei und einem Symbol aus den VFP Systemressourcen gewählt werden.



Wenn einem Menüeintrag ein Symbol zugeordnet ist und sich dieser Menüeintrag wie ein Kontrollkästchen verhalten soll, dient das Symbol als Markierung. Wenn der Eintrag markiert wird, erscheint das Symbol eingedrückt. Wenn die Markierung aufgehoben wird, erscheint das Symbol normal.

Die Position der einzelnen Einträge innerhalb des Menüs kann per drag & drop verändert werden. Für diesen Vorgang ist der Vierwegepfeil ✥, der sich links neben allen Einträgen befindet, festzuhalten. In einigen Fällen sind Verschiebeoperationen nicht möglich. Ein Menü-Pad kann nicht in einen Menüeintrag umgewandelt werden und umgekehrt. Außerdem ist es nicht möglich einen Menüeintrag in ein Untermenü zu verschieben.

Weitere Eigenschaften der Menüeinträge können im unteren Teil des Menü-Designers eingestellt werden. Dazu gehören der Zeichensatz, die Vordergrund- und Hintergrundfarbe, eine Meldung, die in der VFP-Statusbar angezeigt wird sowie der Name einer Konstanten, die verwendet wird, wenn ein mehrsprachiges Menü erstellt wird. Alle Änderungen werden unmittelbar im aktiven Element angezeigt.

Mit der Schaltfläche *ActionOnSelect* kann in einem Editor-Fenster die auszuführende Aktion eingegeben werden. Über die Schaltfläche *SkipFor* kann eine Bedingung eingegeben werden. Wenn diese Bedingung .T. liefert, kann der dazugehörige Menüeintrag nicht ausgewählt werden.

Die eingestellten Eigenschaften beziehen sich immer auf den aktiven Menüeintrag. Neue Menüeinträge erben die Eigenschaften des zuvor ausgewählten Eintrags.

Der Zeichensatz kann über die Schaltfläche *Font* ausgewählt werden. Der Standard-Windows-Dialog zur Auswahl eines Zeichensatzes erscheint. In diesem Dialog können insbesondere die Schriftart und die Schriftgröße sowie der Schriftschnitt ausgewählt werden.

Zu jeder Zeit kann eine Vorschau des Menüs angezeigt werden, indem in der Symbolleiste oder im VMD-Menü *Preview* gewählt wird.

Der VMD erstellt nach der Bearbeitung eines Menüs automatisch die erforderlichen Include-Dateien für sprachunabhängige Menüs. Zusätzliche Arbeitsschritte nach der Bearbeitung von Menüs sind nicht erforderlich.

9. Bedienung und Eigenschaften für Endbenutzer

Die mit den VFX – Formularassistenten erstellten Formulare haben standardmäßig viele gute Eigenschaften. Die Position des Formulars auf dem Bildschirm, die Größe des Formulars (die Größe eines Formulars kann mithilfe eines Resizers vom Benutzer zur Laufzeit eingestellt werden), die zuletzt aktive Seite des Seitenrahmens sowie die Einstellungen des Grid Sortierfolge, Spaltenbreiten, werden für jeden Benutzer individuell gespeichert. Schließt ein Benutzer ein Formular und öffnet er es wieder, erscheint es genauso, wie er es verlassen hat.

9.1. Formularbedienung CDataFormPage

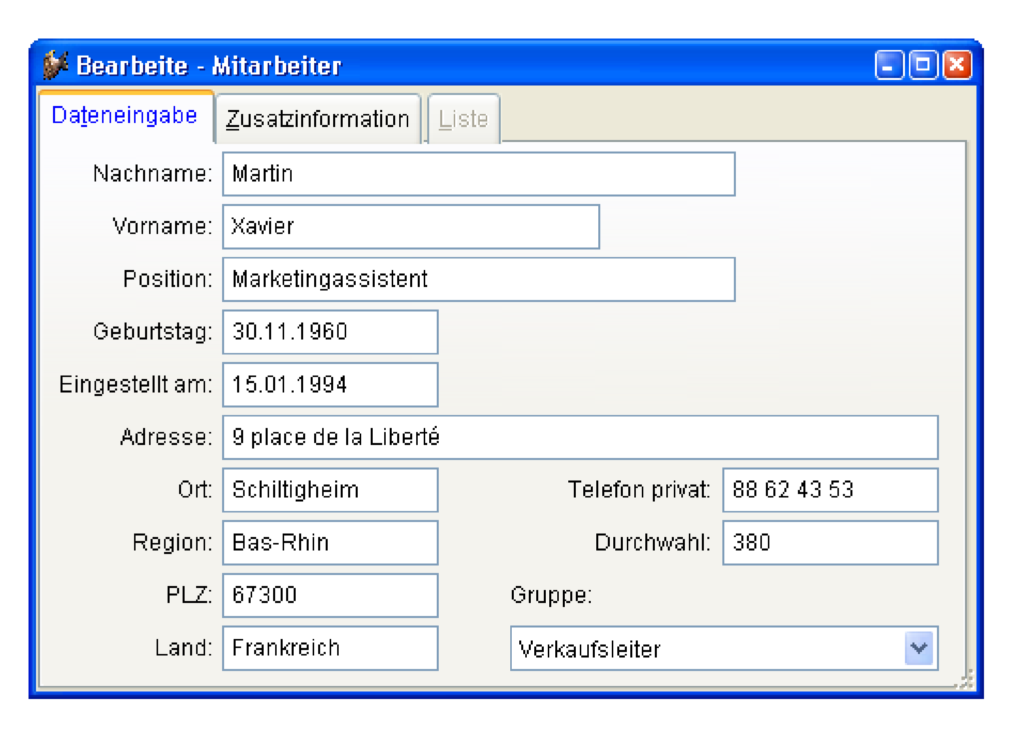
Die Standardbedienung für ein Standard-Datenbearbeitungsformular sieht wie folgt aus, wenn Sie sich nicht im Bearbeitungsmodus oder im Einfügemodus befinden:



Wenn Sie sich im Einfüge- oder Bearbeitungsmodus befinden, ändert sich die Überschrift des Formulars und die Schaltflächen der Symbolleiste werden entsprechend aktualisiert.

ANMERKUNG: Um große Datenmengen einzugeben, können Sie die Tastenkombination *Strg+N* drücken, auch wenn Sie sich bereits im Einfügemodus befinden. Dadurch ist es sehr schnell, mehrere Datensätze nacheinander zu erfassen. Aus den gleichen Optimierungsgründen bleiben die Navigations-Schaltflächen auch während der Bearbeitung aktiv.

Entsprechend der Einstellung der Eigenschaft *nAutoEdit* im Anwendungsobjekt bzw. der Formulareigenschaft *lAutoEdit* kann der Benutzer einfach mit der Bearbeitung beginnen und das Formular wechselt automatisch in den Bearbeitungsmodus, wie hier gezeigt wird:

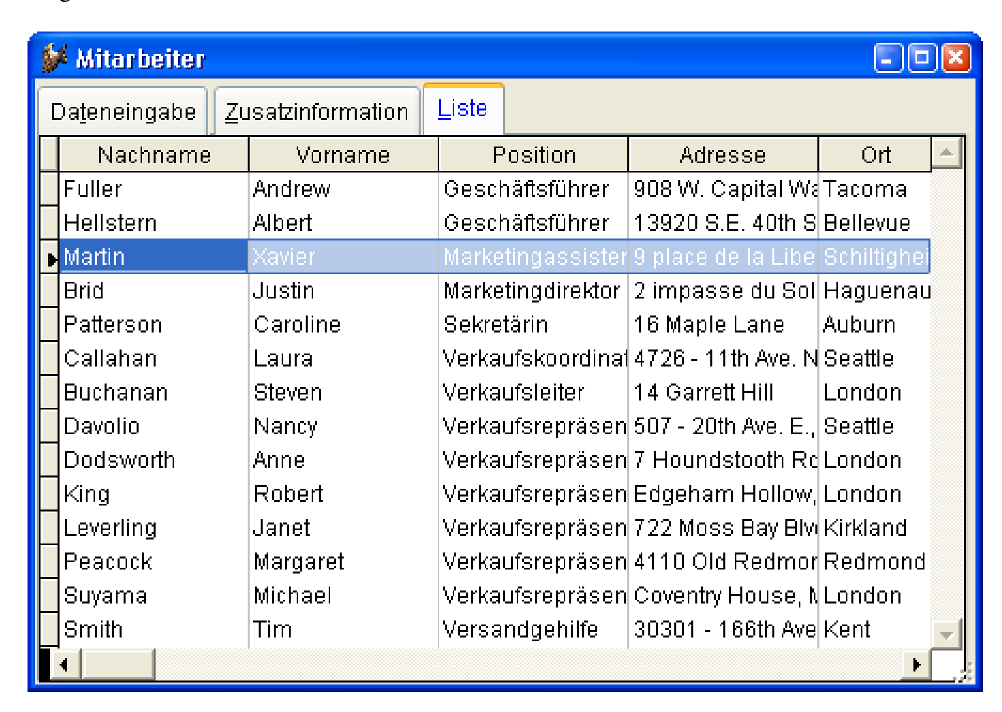


Die Schaltflächen der Symbolleiste sowie die Menüeinträge werden entsprechend dem Formularstatus aktiviert.

9.2. Das VFX Power Grid

In allen Spalten eines Grid ist standardmäßig eine inkrementelle Suche möglich. Durch einen Doppelklick auf eine Überschrift in einem Grid kann die entsprechende Spalte sortiert werden. Wenn für die Spalte kein geeigneter Index vorhanden ist, wird von VFX automatisch ein temporärer Index angelegt. Die temporäre Indexdatei wird gelöscht, wenn das Formular geschlossen wird.

Soll die Suche um eine zusätzliche Spalte erweitert werden, drückt man die Taste „Strg“ und klickt gleichzeitig auf eine weitere Überschrift. Die Rangfolge der Sortierung wird in den Überschriften durch Zahlen in Klammern dargestellt.



Nachname	Vorname	Position	Adresse	Ort
Fuller	Andrew	Geschäftsführer	908 W. Capital Wa	Tacoma
Hellstern	Albert	Geschäftsführer	13920 S.E. 40th S	Bellevue
Martin	Xavier	Marketingassister	9 place de la Libe	Schiltighei
Brid	Justin	Marketingdirektor	2 impasse du Sol	Haguenau
Patterson	Caroline	Sekretärin	16 Maple Lane	Auburn
Callahan	Laura	Verkaufskoordinat	4726 - 11th Ave. N	Seattle
Buchanan	Steven	Verkaufsleiter	14 Garrett Hill	London
Davolio	Nancy	Verkaufsrepräsen	507 - 20th Ave. E.,	Seattle
Dodsworth	Anne	Verkaufsrepräsen	7 Houndstooth Rc	London
King	Robert	Verkaufsrepräsen	Edgeham Hollow,	London
Leverling	Janet	Verkaufsrepräsen	722 Moss Bay Blv	Kirkland
Peacock	Margaret	Verkaufsrepräsen	4110 Old Redmor	Redmond
Suyama	Michael	Verkaufsrepräsen	Coventry House, M	London
Smith	Tim	Versandgehilfe	30301 - 166th Ave	Kent

Ein Doppelklick auf eine Überschrift sortiert eine Spalte. Ein weiterer Doppelklick kehrt die Sortierfolge um.

Nach einem Klick in eine Spalte kann mit der Eingabe eines Suchbegriffs begonnen werden. Die Sortierfolge wird auf diese Spalte umgestellt und der eingegebene Begriff wird inkrementell gesucht. Der eingegebene Begriff wird in der Statuszeile angezeigt:

Suche : Martin

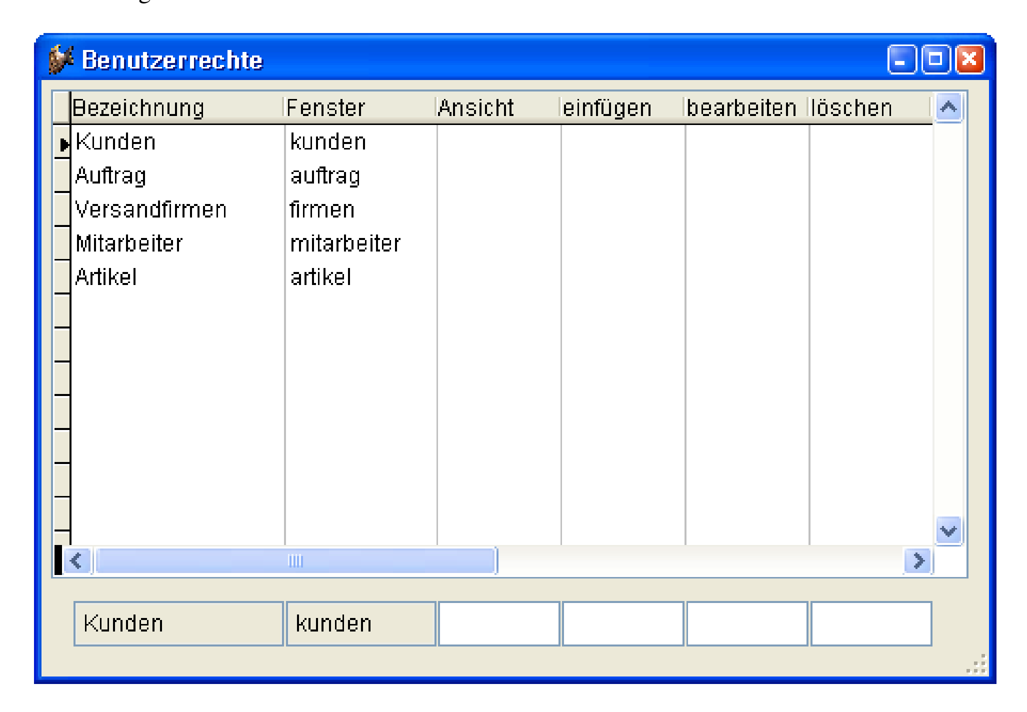
Benutzen Sie den VFX – CGrid Builder, um einzustellen für welche Spalten die inkrementelle Suche verwendet werden soll. Dadurch erhält der Benutzer die Möglichkeit, durch einfaches Eingeben eines Zeichens, einer Zahl oder auch eines Datums die inkrementelle Suche einzuleiten. Dabei wird die Sortierfolge automatisch umgestellt und es wird auf den der Eingabe entsprechenden Eintrag gesprungen. Während der inkrementellen Suche, wird der Suchbegriff in der Statuszeile angezeigt. Korrekturen können mit der Rückschritttaste durchgeführt werden.

VFX zeigt die aktuelle Sortierfolge in der Spaltenüberschrift des Grids an. Der Entwickler kann aus den folgenden Anzeigemöglichkeiten auswählen:

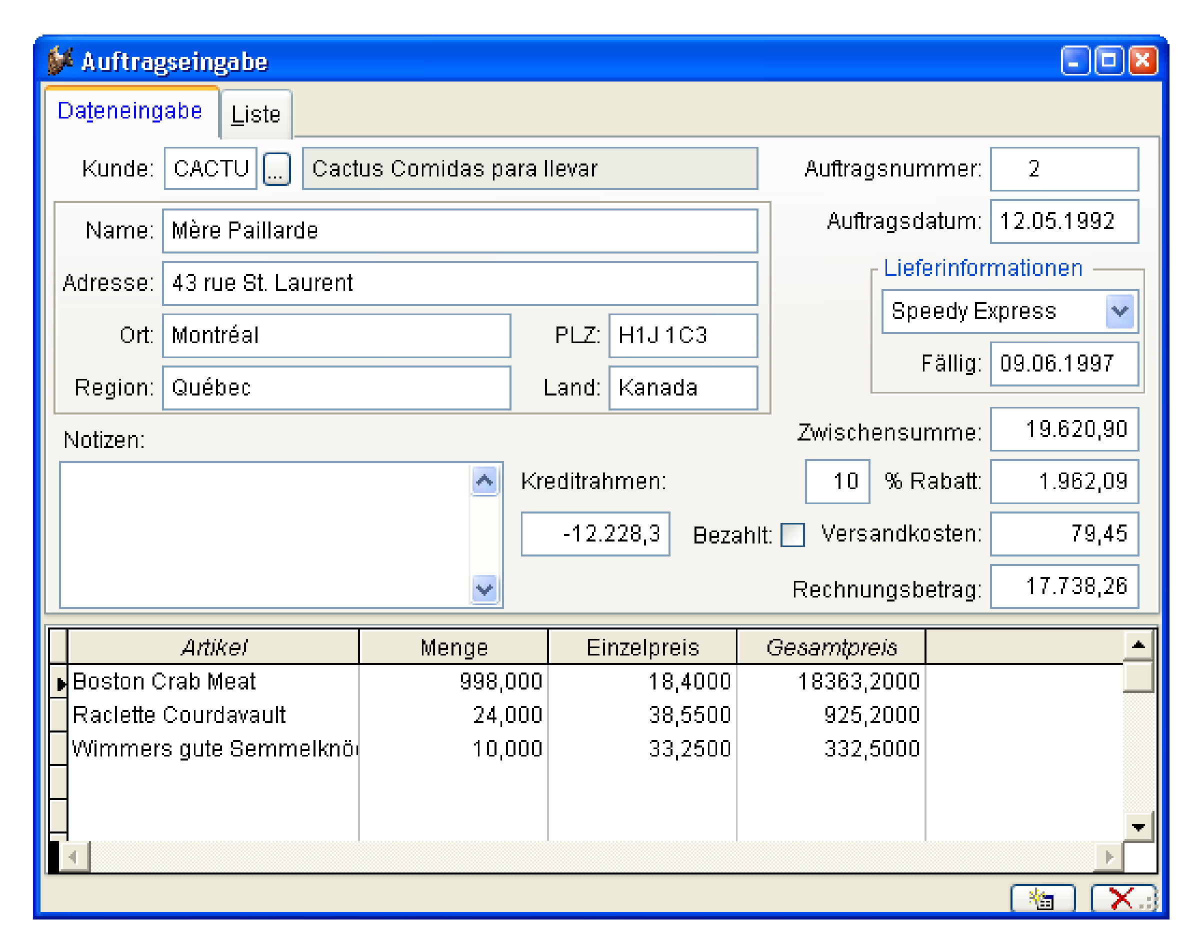
- Keine Anzeige
- Unterstrichene Überschrift
- Anzeige durch verschiedene Farben
- Anzeige durch einen auf- oder absteigenden Pfeil, ähnlich dem Windows-Explorer

9.3. Formularbedienung CTableForm

Bei Formularen basierend auf der Klasse *CTableForm* sind das Such-Grid und andere Steuerelemente nebeneinander oder untereinander auf einem Container angeordnet. Ein typisches *CTableForm*-Formular ist die Verwaltung der Benutzerrechte.



9.4. Formularbedienung COneToManyForm



Die Bearbeitung der Daten der Haupttabelle ist identisch mit der im Standard-Datenbearbeitungs-Formular. Die Symbolleiste und das Menü *Bearbeiten* beziehen sich auf die Haupttabelle.

Die Child-Datensätze werden im unteren Grid bearbeitet. Nur wenn Sie sich im Bearbeitungs- oder Einfügemodus der Haupttabelle befinden, können Sie auch das Child-Grid bearbeiten, Child-Datensätze einfügen und löschen. Alle Bearbeitungen der Child-Datensätze werden mit optimistischer Tabellenpufferung durchgeführt. Wenn Sie sich entscheiden, Ihre Änderungen rückgängig zu machen, werden die Änderungen in allen Child-Datensätzen rückgängig gemacht. Wenn Sie sich entscheiden, die Änderungen zu speichern, werden alle Änderungen an der Haupttabelle und in allen Child-Datensätzen gespeichert.

Ein Klick in den leeren Bereich eines Child-Grids fügt einen neuen Child-Datensatz an.

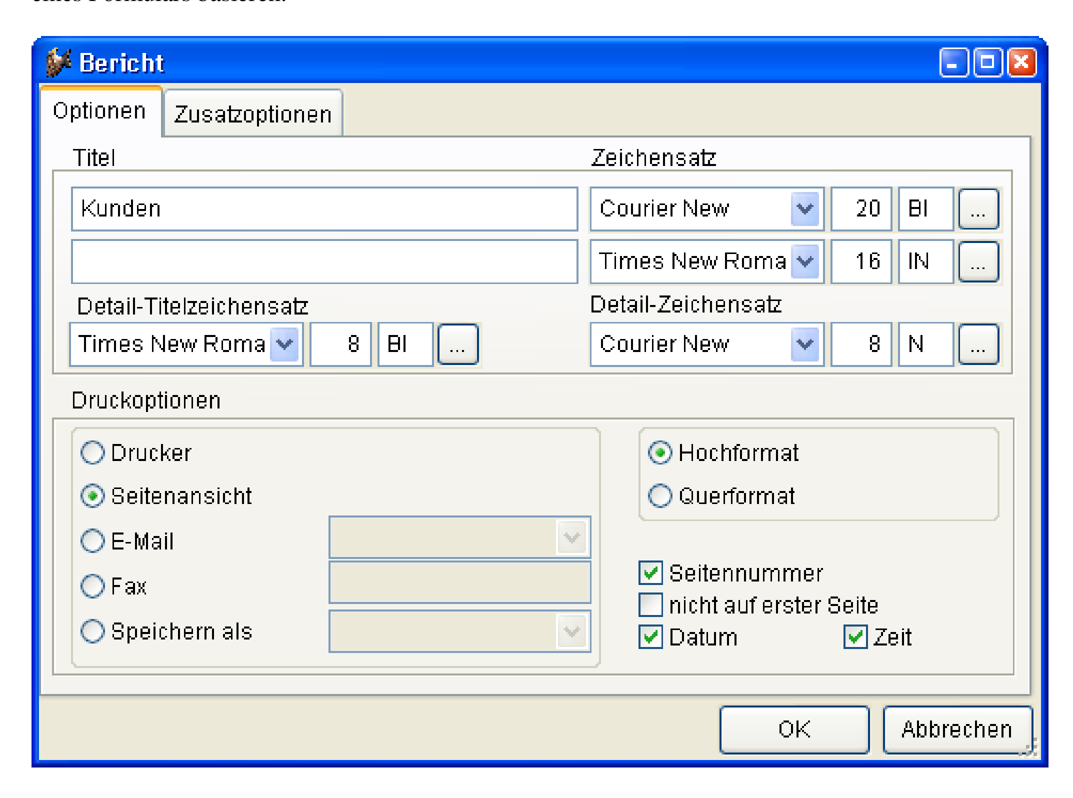
Wenn die Child-Daten auf einer Ansicht oder auf einem Cursoradapter basieren, kann in den Child-Daten inkrementell gesucht werden.

Eine der interessantesten Funktionen von VFX ist die besondere Auswahlliste, die Sie Ihrem Child-Grid mit dem VFX – CPickTextBox Builder hinzufügen können. Die Auswahllisten können im Bearbeitungs- und im Einfügemodus erreicht werden.

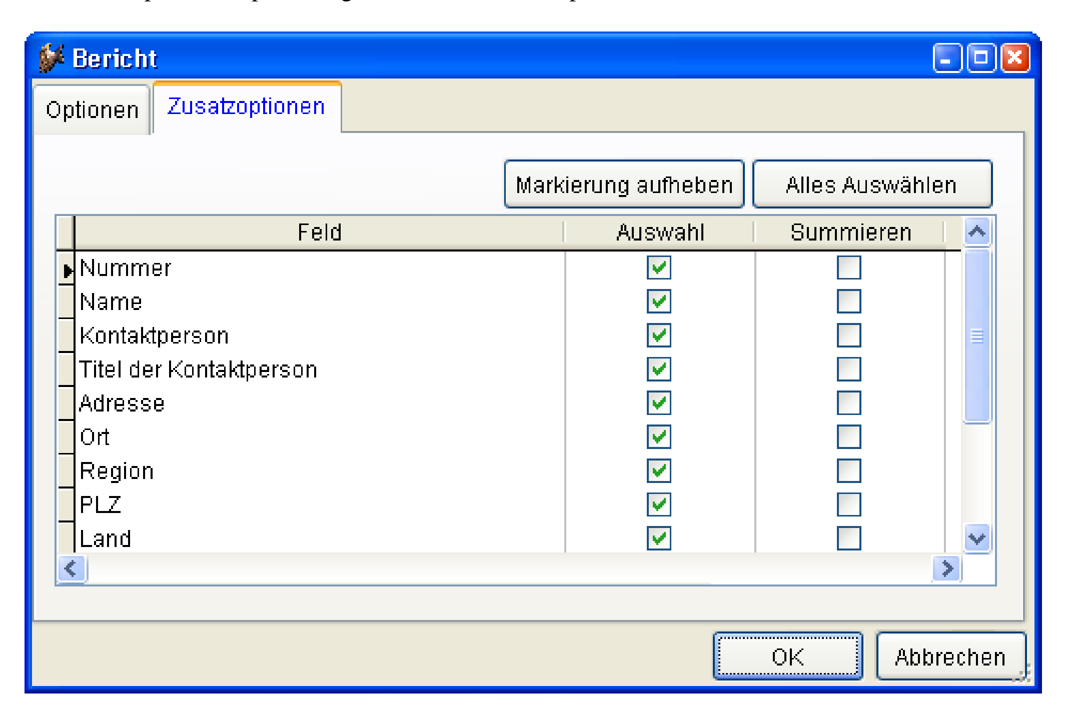
Durch einen Doppelklick in die *CPickTextBox* oder durch drücken der Funktionstaste *F9* wird die Auswahlliste angezeigt.

9.5. Drucken

Aus allen Formularen kann standardmäßig eine Liste gedruckt werden, ohne dass dafür Berichte angelegt werden müssen. VFX legt zur Laufzeit der Anwendung temporäre Berichtsdateien an, die auf der Ansicht der Suchseite eines Formulars basieren.

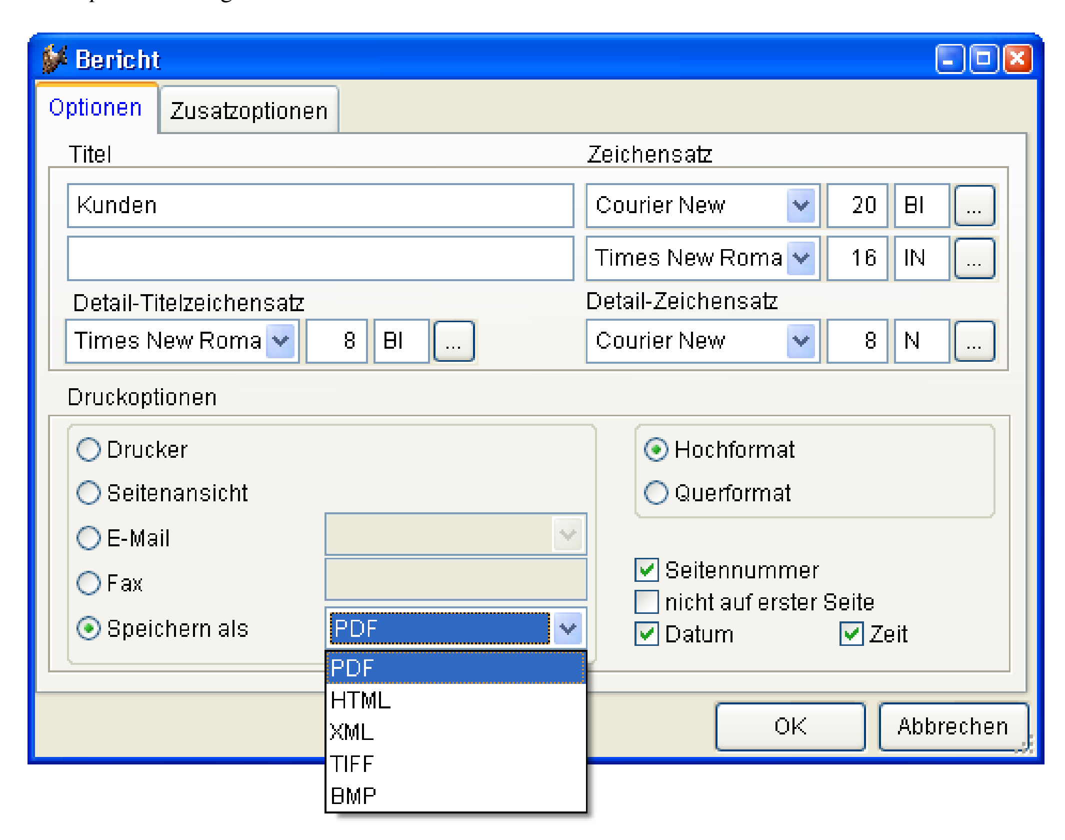


Vor dem Druck bzw. der Seitenansicht kann der Benutzer nicht gewünschte Spalten aus der Liste entfernen. Die Breite der Spalten entspricht ungefähr der Breite der Spalte im Grid.



VFX 11.0 unterstützt alle Möglichkeiten von VFP 9 um Berichtsausgaben in verschiedenen Dateiformaten speichern zu können. Die unterstützen Dateiformate sind PDF, HTML, XML, TIFF and BMP. Alle diese Dateiformate können auch als E-Mailanhang versendet werden.

Im Berichtsdialog kann das Dateiformat in einer Combobox ausgewählt werden, wenn eine der Optionen *E-Mail* oder *Speichern als* gewählt wird.

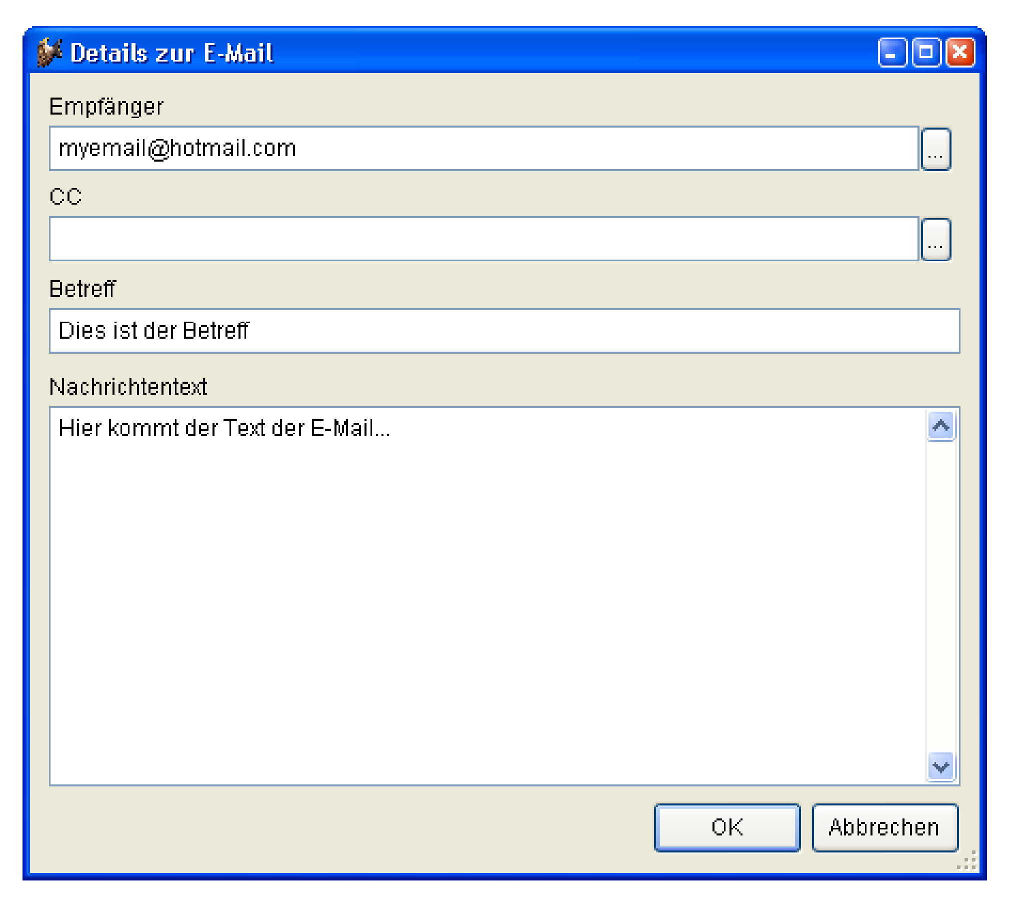


Wenn als Dateiformat TIFF oder BMP gewählt wird, wird für jede Seite des Berichts eine eigene Datei angelegt. Dem vom Anwender eingegebenen Dateinamen wird ein numerischer Wert mit der jeweiligen Seitennummer angehängt.

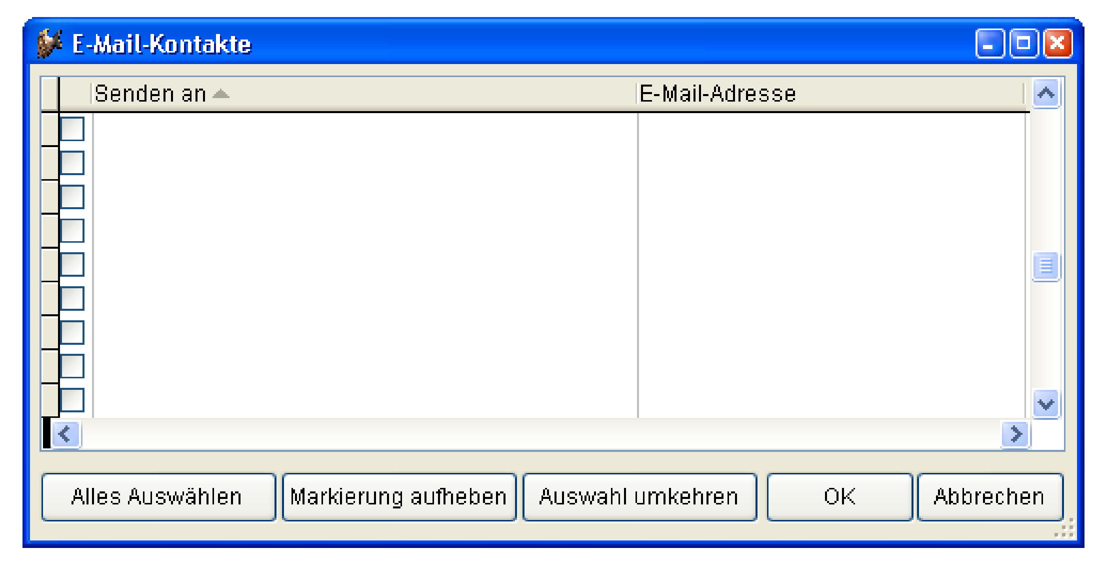
9.6. E-Mailversand

Alle Dateiformate, in denen Berichtsausgaben gespeichert werden können, können als E-Mailanhang versendet werden.

Im Dialog *Details zur E-Mail* können ein oder mehrere E-Mailempfänger, CC-Empfänger, der Betreff und ein Text eingegeben werden. Wenn der Wert der Eigenschaft *goProgram.lUseBCCRecipients* auf *.T.* eingestellt ist, können auch BCC-Empfänger eingegeben werden.



Für jede Art von Empfängerliste kann über eine Schaltfläche eine Auswahlliste mit allen Adressen aus dem Outlook-Adressbuch angezeigt werden.

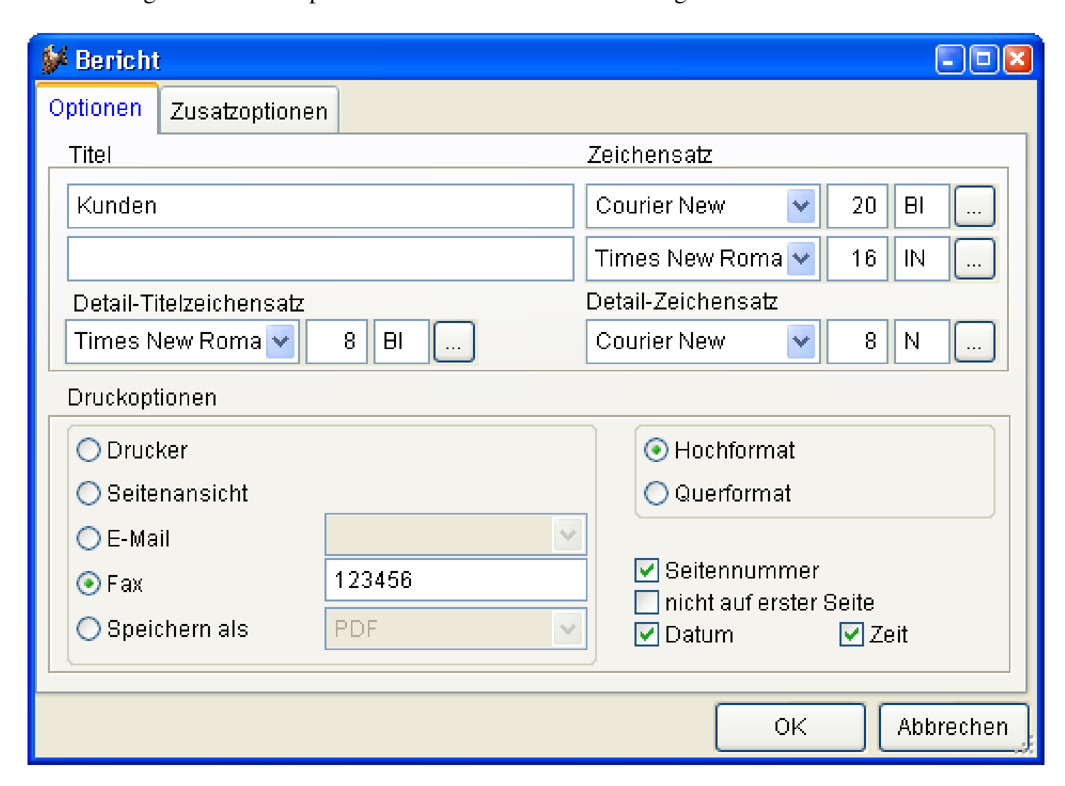


Die ausgewählten E-Mailadressen werden durch einen Klick auf die Schaltfläche *OK* in das Feld mit der Empfängerliste übernommen.

9.7. Faxversand

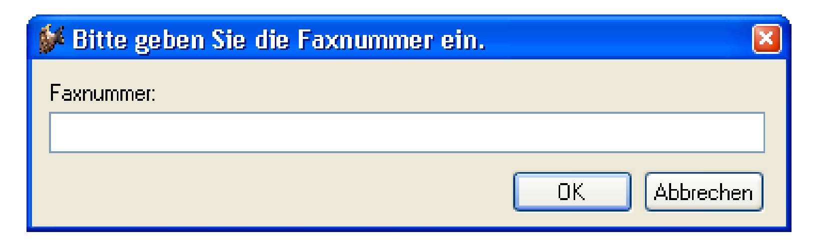
Eine weitere Möglichkeit Berichtsausgaben zu erzeugen ist der Versand als Fax. Wenn der Anwender die Fax-Option wählt, muss die Faxnummer eingegeben werden.

VFX 11.0 unterstützt die Fax-Programme FRITZ!fax von AVM und Winfax von Symantec. VFX 11.0 erkennt automatisch, ob eins dieser beiden Fax-Programme installiert ist. Wenn ein Fax-Programm erkannt wird, wird die Berichtsausgabe an den entsprechenden Fax-Druckertreiber übergeben.



Die Faxnummer wird von der VFX-Anwendung direkt an das Fax-Programm übergeben. Der Endanwender wird nicht mit Dialogen des Fax-Programms konfrontiert.

Wenn einem Formular eine individuelle Berichtsdatei zugeordnet ist, kann der Anwender die Faxnummer im abgebildeten Dialog eingeben:

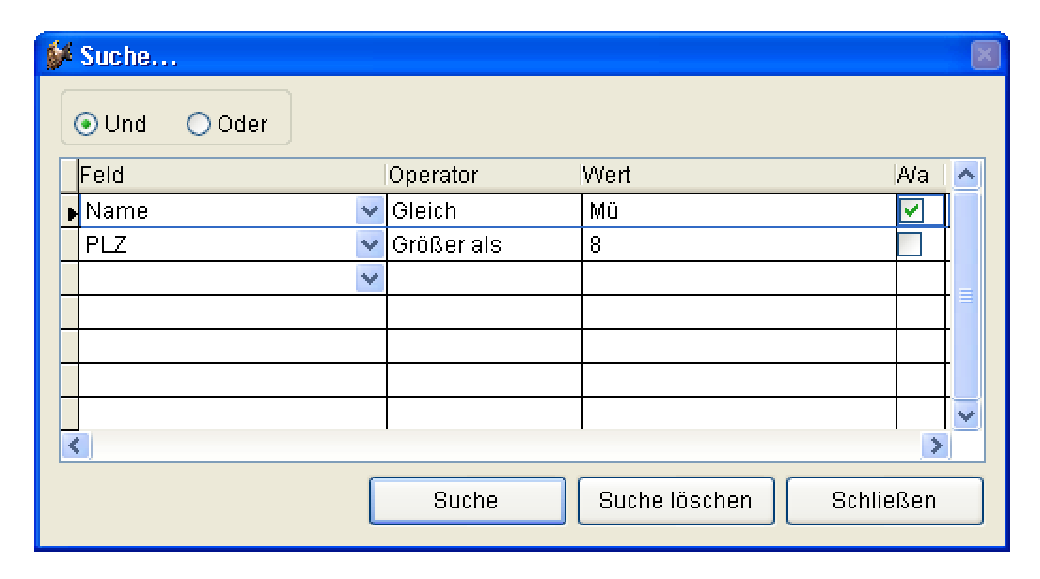


9.8. Suchen

Der sichtbare Datenbereich in einem Formular kann durch Setzen eines Filters eingeschränkt werden. VFX stellt dafür einen fertigen Dialog zur Verfügung. Beliebig viele Felder können dabei mit „und“ oder „oder“ verknüpft werden.

Es können beliebig viele Suchkriterien kombiniert werden. Die Suchkriterien werden je Benutzer und Formular gespeichert und stehen auch nach einem Neustart des Programms wieder zur Verfügung.

Im Suchdialog können Anwender nur gültige Ausdrücke eingeben. Je nach gewähltem Datentyp stehen nur die geeigneten Vergleichsoperatoren zur Verfügung. Es können nur Werte vom gleichen Datentyp eingegeben werden.



In der Spalte *Wert* befinden sich mehrere Steuerelemente. Die Eigenschaft *CurrentControl* dieser Spalte wird abhängig vom Datentyp des in der Spalte *Feld* gewählten Feldes umgeschaltet.

Wenn ein Feld vom Typ *Zeichen* gewählt wird, wird in der Spalte *Wert* eine Textbox angezeigt, in die beliebige Werte eingegeben werden können. Es steht zusätzlich der Vergleichsoperator *Enthält* zur Verfügung. In diesem Fall wird der Filterausdruck mit dem Operator *$* aufgebaut. Zusätzlich kann in der rechten Spalte im Grid für jede Zeile eingestellt werden, ob die Groß-/Kleinschreibung berücksichtigt werden soll.

Wenn ein numerisches Feld in der ersten Spalte gewählt wird, wird in der Spalte *Wert* eine Textbox angezeigt, die es dem Benutzer erlaubt nur Zahlenwerte einzugeben.

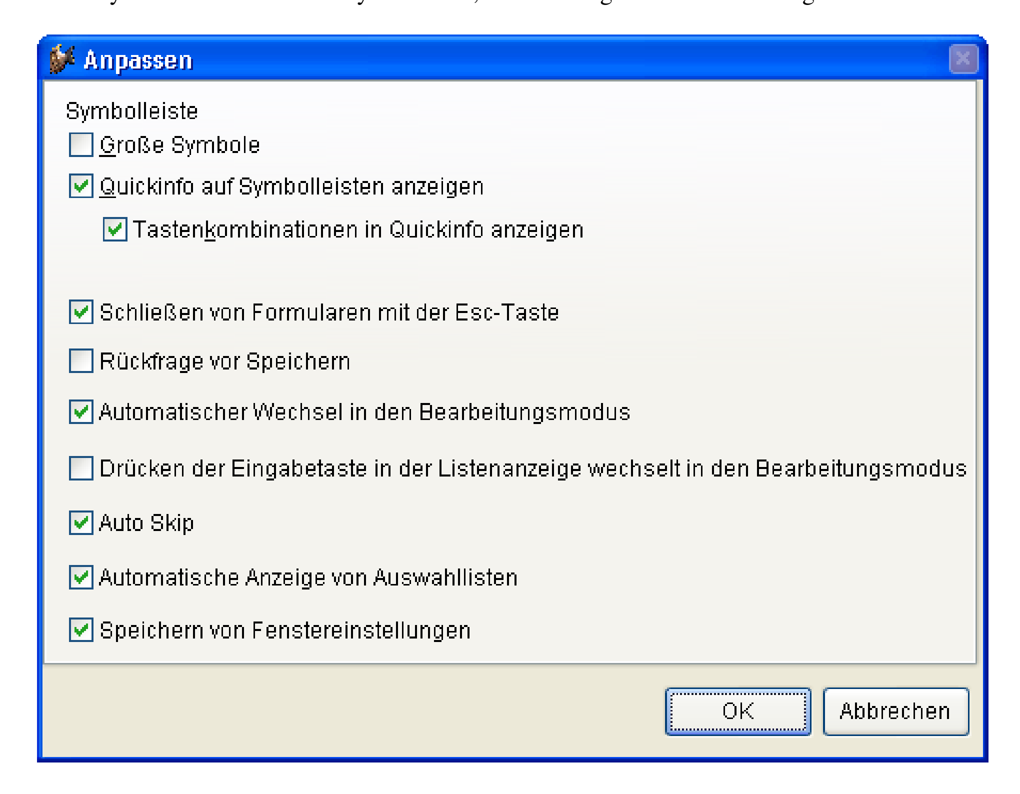
Wenn ein Feld vom Typ *Date* oder *Datetime* gewählt wird, wird die *Inputmask* in der Spalte Wert entsprechend eingestellt.

Wenn ein logisches Feld ausgewählt wird, kann in der Spalte *Wert* in einer Combobox *Wahr* oder *Falsch* ausgewählt werden. Die manuelle Eingabe eines Wertes durch den Anwender ist nicht erforderlich.

Auf diesem Weg ist es dem Benutzer nicht möglich unzulässige Werte in der Spalte *Wert* einzugeben.

9.9. Layout

Das Erscheinungsbild von VFX 11.0-Anwendungen wurde durch neue Symbole im Windows-XP-Stil verbessert. Neue Symbole wurden für die Symbolleiste, Menüeinträge und andere Dialoge entwickelt.



Endbenutzer können das Layout der Anwendung über den Menüpunkt *Extras, Anpassen* selbst entsprechend den eigenen Wünschen einstellen. Es kann zwischen kleinen und großen Symbolen in Symbolleisten gewählt werden. Wahlweise können Quickinfos angezeigt werden. Wenn das Kontrollkästchen *Tastenkombinationen in Quickinfo anzeigen* markiert ist, werden an die Quickinfo die Hotkeys angefügt. Beispielsweise ist der Hotkey für die Schaltfläche *Neu* die Tastenkombination *Strg+N*.

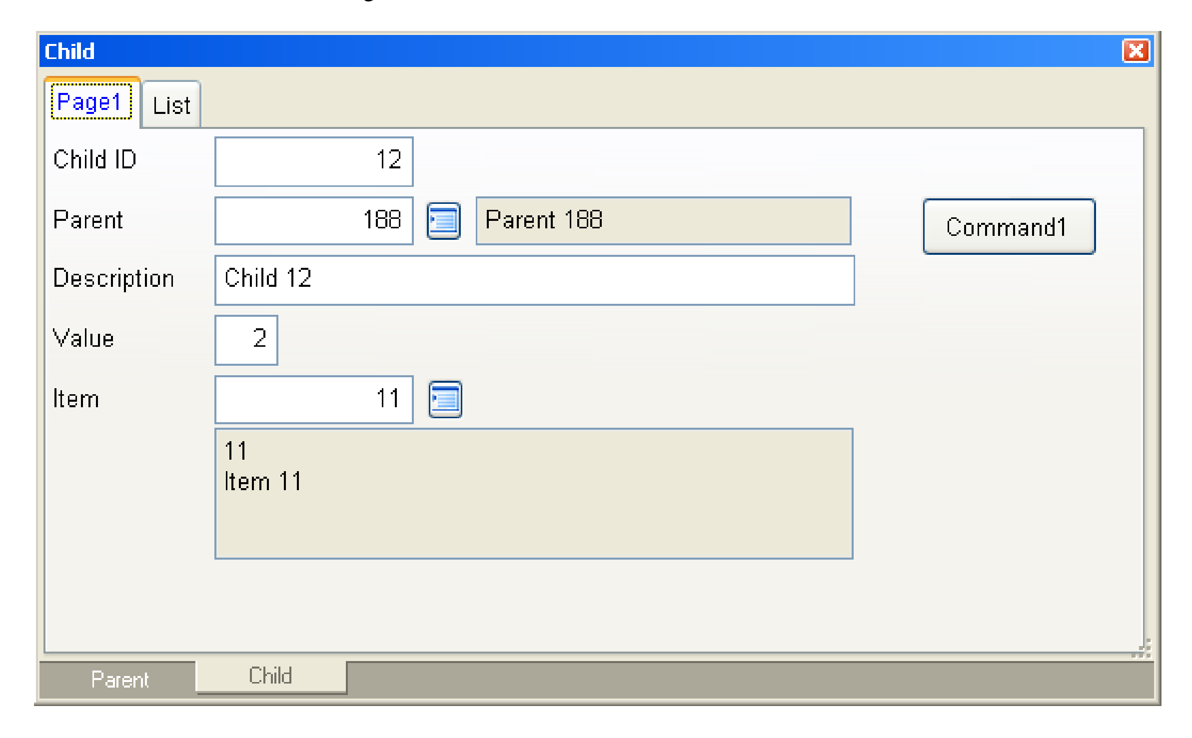


Neu für das Layout von VFX-Formularen ist die Möglichkeit Hintergrundbilder für Seiten auf Seitenrahmen in Formularen auszuwählen. Das Hintergrundbild kann in den VFX Form Buildern eingestellt werden.

Anstelle eines Hintergrundbildes kann mit den VFX – Form Buildern auch eine Hintergrundfarbe für Seiten eines Seitenrahmens eingestellt werden.

9.10. Gedockte Formulare

VFX 11.0 unterstützt ineinander gedockte Formulare.



Das Dock-Verhalten von Formularen wird durch die Eigenschaft *goProgram.nDockable* des Anwendungsobjekts gesteuert. Wenn der Wert dieser Eigenschaft auf -1 eingestellt ist, wird die Einstellung des Formulars verwendet. Wenn *goProgram.nDockable* einen Wert größer als 1 enthält, wird dieser Wert in der Eigenschaft *Dockable* des Formulars gespeichert.

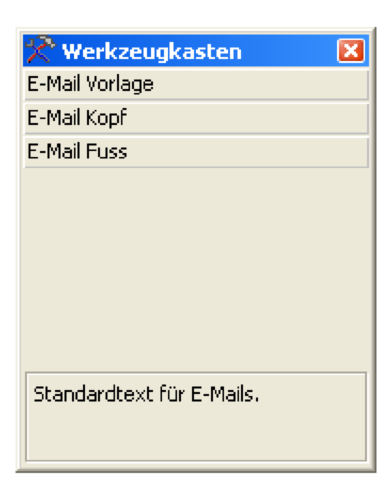
ANMERKUNG: Wenn die Eigenschaft *WindowType* des Formulars auf *Modal* eingestellt ist, wird die Eigenschaft *goProgram.nDockable* nicht ausgewertet. Modale Formulare können grundsätzlich nicht gedockt werden.

Der Dockstatus und die Dockposition eines Formulars werden für jeden Benutzer in der Ressourcentabelle *Vfxres.dbf* gespeichert.

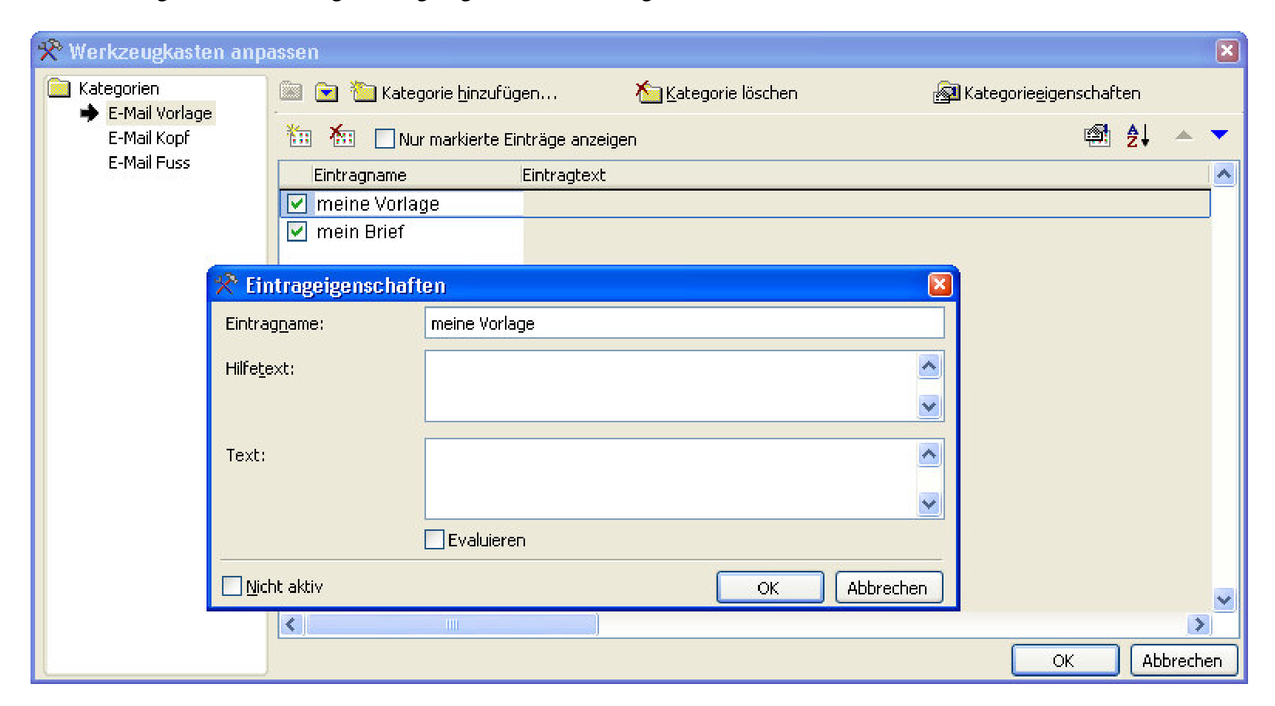
9.11. VFP Toolbox für Endanwender

Die VFP Toolbox ist in VFP 9 auch für Endanwender nutzbar. In VFX 11.0 wurde die Toolbox vollständig integriert und an VFX angepasst. Ähnlich wie die Toolbox für Entwickler, dient der Werkzeugkasten für Endanwender als universelle Drag & Drop Quelle bzw. auch als Ziel. Einträge aus dem Werkzeugkasten können in Textboxen, Editboxen und andere Drop-Ziele gezogen werden.

Die Einträge im Werkzeugkasten sind in Kategorien gruppiert.



Mit einem Rechtsklick auf dem Werkzeugkasten und über den Kontextmenüpunkt *Werkzeugkasten anpassen* können Kategorien und Einträge hinzugefügt, bearbeitet und gelöscht werden.



Für jede Kategorie können der Kategoriename und ein Hilfetext gespeichert werden. Für Einträge können ein Eintragsname, ein Hilfetext und ein Eintragstext gespeichert werden.

Kategorienamen und Eintragsnamen werden im Werkzeugkasten angezeigt. Der jeweilige Hilfetext wird am unteren Rand des Werkzeugkastens in einer Editbox als Beschreibung zum aktuellen Eintrag angezeigt. Der Eintragstext wird auf dem jeweiligen Drop-Ziel eingefügt.

Mit den Schaltflächen und können Anwender die Reihenfolge der Kategorienanzeige im Werkzeugkasten ändern. Einträge können mit den Schaltflächen und innerhalb einer Kategorie verschoben werden

9.12. Treeview

Die Klasse *CTreeView* wurde so verbessert, dass eine wesentliche verkürzte Ladezeit erreicht werden konnte. Der aktuelle Zustand aller Knoten, geöffnet oder geschlossen, wird in der Ressourcentabelle *Vfxres.dbf* für jeden

Benutzer gespeichert. Beim erneuten Öffnen eines Formulars erscheinen alle Knoten in dem Zustand, in dem das Formular geschlossen wurde.

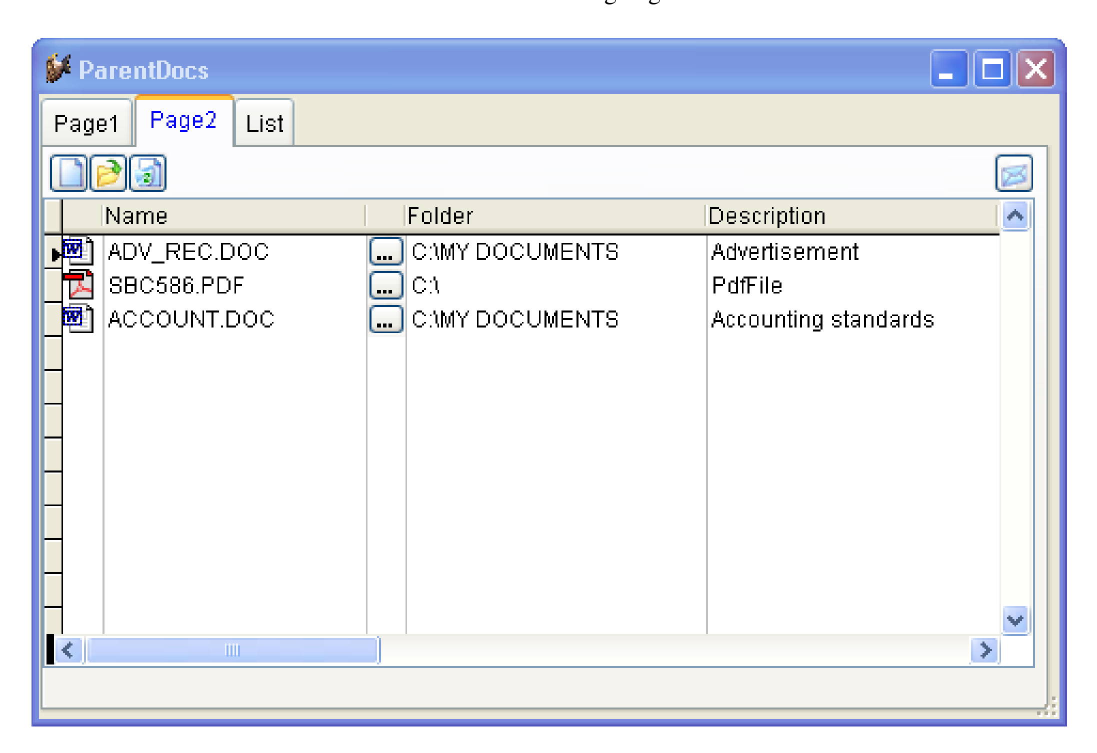
Es ist jetzt möglich aus Formularen basierend auf einer der Formularklassen *CTreeviewForm* oder *CTreeviewOneToMany* Berichte zu drucken, die die Struktur des Treeview beinhalten.

Dem Treeview-Steuerelement wurde ein Kontextmenü mit den Einträgen *Neu*, *Umbenennen* und *Löschen* hinzugefügt.

9.13. Dokumentenverwaltung mit der Klasse CDocumentManagement

Die neue Klasse *CDocumentManagement* dient zur Verwaltung von Dokumenten aller Art (z. B. Word, Excel, Powerpoint) innerhalb einer Anwendung. Die Klasse *CDocumentManagement* ist ein Container, der Child-Datensätze zum aktuellen Datensatz im Formular verwaltet. Die Dokumentenverwaltung ermöglicht dem Anwender Dokumente zu öffnen und als E-Mailanhang zu versenden.

Diese Klasse kann bestehenden Formularen einfach hinzugefügt werden.



9.14. Info-Dialog

Dem Info-Dialog wurde ein Link-Label zur Anzeige des Endbenutzer-Lizenzvertrags (EULA) hinzugefügt. Über dieses Link-Label wird ein Dialog angezeigt, indem der Benutzer den Lizenzvertrag lesen und drucken kann.

Der Endbenutzer-Lizenzvertrag ist in der Tabelle *Vfxinternfiles.dbf* gespeichert. So ist es einfach möglich für jede Sprache einen lokalisierten Endbenutzer-Lizenzvertrag zur Verfügung zu stellen.

9.15. Weitere Verbesserungen für Endbenutzer in VFX 11.0

- Unterstützung der inkrementellen Suche auch wenn der aktuelle Zelleninhalt .NULL. ist.
- Lokalisierte Hotkeys für die Klasse *CPickDate* und ein mehrzeiliger Tooltip als Hilfe.
- Neue Klassen: E-Mail mit Outlook-Aufruf, Hyperlink mit Internet Explorer-Aufruf, numerische Textbox mit Taschenrechneraufruf, TAPI, Dateiauswahl mit Fileselectbox.
- Unterstützung von *visible=.F.* in Grid-Columns für den Suchdialog und den Druckdialog.
- Restzeitanzeige bei der Aktualisierung der Kundendatenbank.
- Skript für Download und Installation von Adobe Reader (für PDF-Dokumente).
- Tastaturbedienung des XP-Öffnen-Dialogs.
- Unterstützung von Drag & Drop in Mover-Dialogen.
- Beim erneuten Öffnen eines Formulars wird der Satzzeiger auf den zuletzt angezeigten Datensatz positioniert.
- Unterstützung der Eigenschaft *HighLightStyle* in Grids.
- Verbesserte Anzeige von Memo-Feldern in Grids.
- Wenn alle Favoriten gelöscht werden, wird das dazugehörige, leere Menü gelöscht.

10. Datenzugriff

10.1. Konzept des Datenzugriffs

Eine der größten Neuerungen in VFX 11.0 ist das völlig neue Konzept des Datenzugriffs. Keine Sorge, bestehende Anwendungen sind mit dem neuen Konzept voll kompatibel. Wie bisher, kann auch weiterhin direkt mit Tabellen oder Ansichten auf lokalen oder Remote-Datenquellen gearbeitet werden.

Der neue Datenzugriff ist vielmehr eine zusätzliche Möglichkeit um auf Daten zuzugreifen. VFX 11.0 unterstützt die VFP-Klasse CursorAdapter beim Zugriff auf Daten. Die VFP-Klasse CursorAdapter kann als kleine Revolution beim Datenzugriff aus VFP-Anwendungen betrachtet werden.

Bisher lief der Datenzugriff in VFP und VFX immer mithilfe eines DBC. Versierte Programmierer konnten auch per SQL Pass Through auf Daten zugreifen, aber das wollen wir hier nicht näher betrachten. Der Zugriff auf Daten mittels eines DBC ist uns gut vertraut und stabil und zuverlässig. Der Datenzugriff auf einen DBC hat aber auch ein paar Nachteile. Ein DBC ist nichts anderes als eine Tabelle. Die Namenserweiterung ist von DBF in DBC geändert, weil es sich um eine besondere Tabelle handelt. Im DBC befinden sich Informationen über die Struktur und die Integrität der Datenbank, aber auch Informationen über Verbindungen, wenn mit Remote-Datenquellen gearbeitet wird.

Anwender könnten den DBC manipulieren. Verbindungsinformationen zu Remote-Datenquellen inklusiv Benutzername und Kennwort sind im Klartext lesbar, wenn der DBC zum Beispiel mit Excel geöffnet wird.
Der Idee ohne DBC arbeiten zu wollen, liegen zwei Erkenntnisse zugrunde. Die Verbindungsinformationen müssen vor unerlaubten Zugriff und Manipulation besser geschützt werden. Die Portierung einer Anwendung von DBC zu einer Remote-Datenquelle soll wesentlich einfacher möglich werden.

Genau diese Ziele können bei Verwendung von CursorAdaptern erreicht werden. CursorAdapter können der Datenumgebung genau wie Tabellen oder Ansichten hinzugefügt werden. CursorAdapter sind Klassen und können vererbt werden. VFX bietet in der Klassenbibliothek *Vfxctrl.vcx* die Klasse *CBaseDataAccess*, die die Grundlage für alle in VFX-Anwendungen verwendeten CursorAdapter bilden sollte.

In Formularen, die als Datenquelle CursorAdapter verwenden, stehen alle guten Eigenschaften von VFX-Formularen, wie inkrementelle Suche in Grids, Filter- und Druckmöglichkeiten, zur Verfügung. Auch die Builder von VFX unterstützen CursorAdapter genauso wie Tabellen oder Ansichten.

CursorAdapter basierend auf *CBaseDataAccess* verwenden den Verbindungs-Manager, den wir schon aus früheren VFX-Versionen kennen, um auf Datenbanken zuzugreifen. Dadurch ist sichergestellt, dass alle CursorAdapter einer Anwendung die gleiche Verbindung benutzen. Dies ist nicht nur eine Optimierung von Ressourcen, sondern ist bei einigen Datenbanken auch aus lizenzrechtlichen Gründen erforderlich, wenn je Verbindung eine Zugriffslizenz benötigt wird.

Die Verbindungsinformationen, die der Verbindungs-Manager verwendet, werden aus der Datei *Config.vfx* gelesen. Ähnlich wie in einem DBC eine Verbindung gespeichert werden kann, können in der Datei *Config.vfx* Verbindungsinformationen zu mehreren Datenbanken gespeichert werden. Die Verbindung kann zu einem DBC oder zu einer Remote-Datenquelle mittels eines DSN-Eintrags oder einer Verbindungszeichenfolge hergestellt werden. Um die Datei *Config.vfx* vor Manipulationen zu schützen, ist sie mit einem Kennwort verschlüsselt. Das zur Entschlüsselung benötigte Kennwort ist in der Eigenschaft *goProgram.cconfigpassword* gespeichert und somit in der kompilierten Exe-Datei enthalten.

Durch einen anderen Eintrag in der Datei *Config.vfx* kann eine bestehende Anwendung von einer Datenquelle zur Verwendung einer anderen Datenquelle umgeschaltet werden. Die Datei *Config.vfx* kann mehrere Verbindungen enthalten. Wenn mehr als eine Verbindung gespeichert ist, erhält der Anwender beim Programmstart einen Auswahldialog. Diese Eigenschaft ist vergleichbar mit der Möglichkeit mehrere Datenbanken in der Tabelle *Vfxpath.dbf* einzutragen, wie wir es aus früheren VFX-Versionen kennen.

10.2. Konzeption neuer Anwendungen

Wer eine neue Anwendung mit VFX 11.0 entwickeln will, sollte das neue Konzept des Datenzugriffs ernsthaft in Erwägung ziehen. Wenn der Datenzugriff einer VFX 11.0-Anwendung ausschließlich über CursorAdapter basierend *CBaseDataAccess* durchgeführt wird, ist die Portierung auf eine andere Datenquelle später problemlos möglich.

So kann eine Anwendung zunächst mit einem DBC als Datenquelle begonnen werden. Mit dem VFX – CursorAdapter Wizard werden dann für alle im DBC enthaltenen Tabellen CursorAdapter angelegt. Diese CursorAdapter werden dann als Datenquelle in allen Formularen verwendet.

10.3. VFX – CursorAdapter Wizard

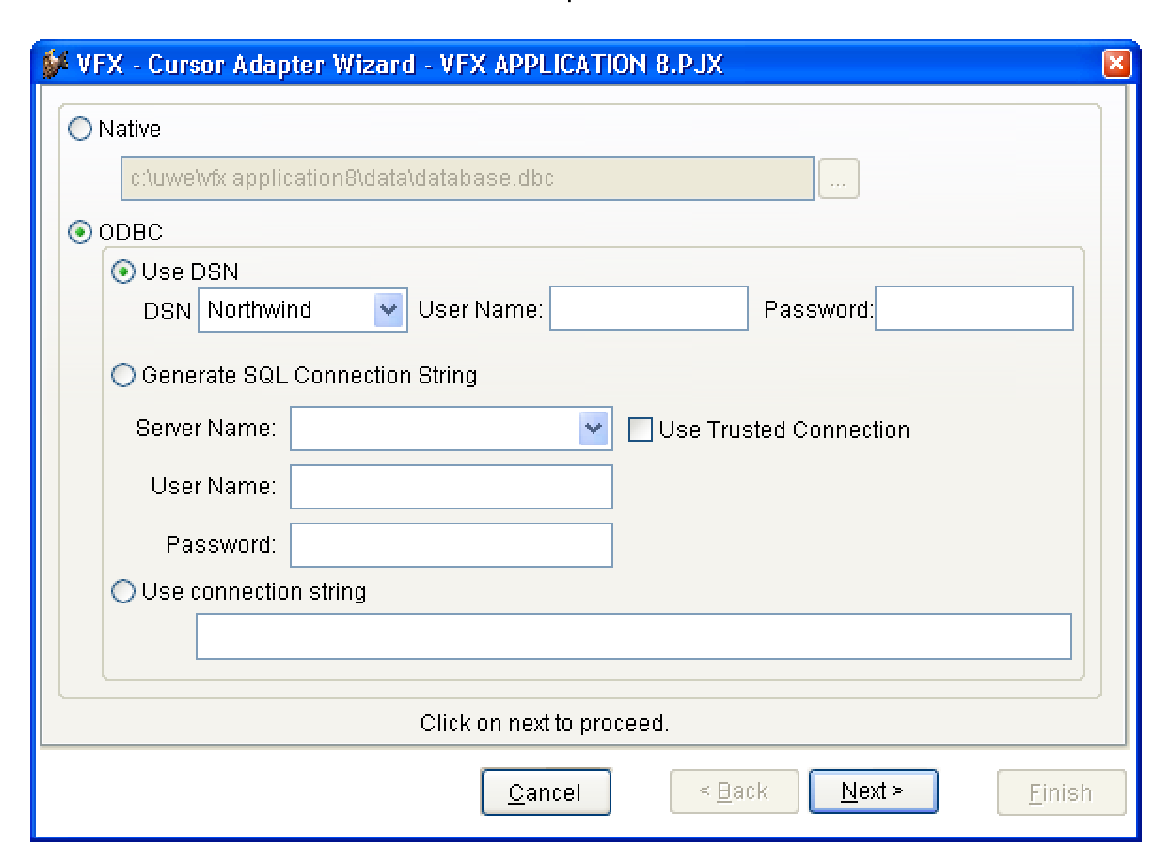
Der VFX – CursorAdapter Wizard erstellt zu jeder Tabelle einer Datenbank eine CursorAdapter-Klasse. Mithilfe der so generierten CursorAdapter kann zum Beispiel aus Formularen auf die Daten zugegriffen werden. Der CursorAdapter Wizard kann eine beliebige von VFP unterstützte Datenquelle als Grundlage zur Generierung von CursorAdaptern verwenden.

Die generierten CursorAdapter-Klassen können nach der Generierung durch den Wizard im VFP Klassen-Designer weiter bearbeitet werden. Es sollte insbesondere in Erwägung gezogen werden, welche Parameter für die CursorAdapter sinnvoll eingesetzt werden können.

Standardmäßig basieren diese CursorAdapter-Klassen auf der Klasse *CAppDataAccess* und werden in der Klassenbibliothek *Appl.vcx* gespeichert. Die Klassenbibliothek und die Basisklasse können bei Bedarf im Wizard geändert werden.

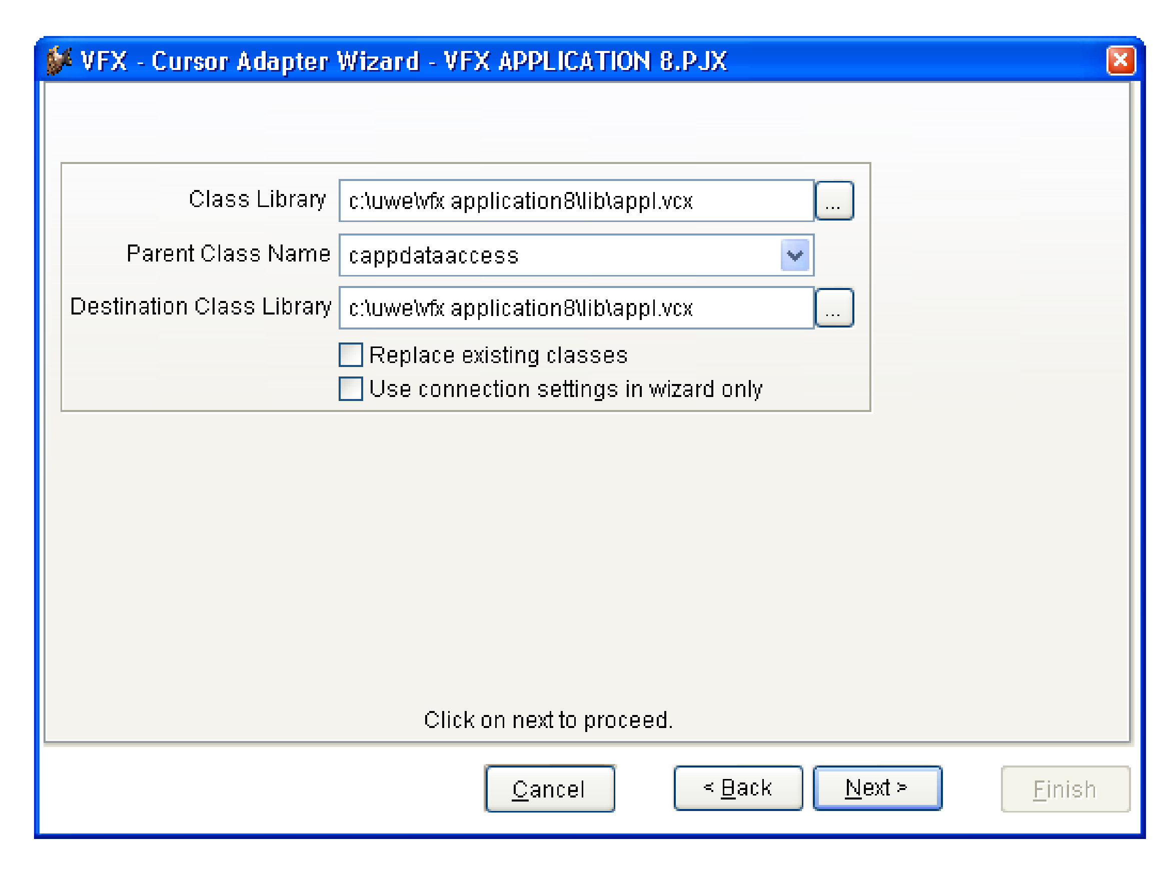
Der Wizard führt den Entwickler durch drei Schritte.

10.3.1. Auswahl der Datenquelle



Diese Datenquelle wird die Datenquelle der Anwendung. Diese Datenquelle wird vom Wizard nur zur Erstellung der CursorAdapter verwendet. Die zur Laufzeit verwendete Datenquelle wird aus der Datei *Config.vfx* gelesen. Auf diesem Weg können für verschiedene Kunden unterschiedliche Datenquellen verwendet werden.

10.3.2. Auswahl der Klassen und Klassenbibliotheken



Wenn die Option *Generate SQL Connection String* gewählt wird, muss im zweiten Schritt zunächst eine Datenbank vom gewählten SQL Server gewählt werden.

In diesem Schritt werden die verwendete CursorAdapter-Basisklasse und die Klassenbibliothek ausgewählt, in der die CursorAdapter gespeichert werden sollen.

Die Standardwerte sind:

Class Library: *Appl.vcx*

Parent Class Name: *CAppDataAccess*

Destination Class Library: *Appl.vcx*

Wahlweise können existierende Klassen in der Zielklassenbibliothek überschrieben werden, wenn eine Markierung im Kontrollkästchen *Replace existing classes* gesetzt wird.

10.3.3. Auswahl der Tabellen

Der letzte Schritt zeigt Listen aller Tabellen und Felder für die CursorAdapter erstellt werden sollen. Beim Bewegen des Satzzeigers in der Tabellenliste auf der linken Seite werden auf der rechten Seite die dazugehörigen Felder angezeigt.

Schlüsselfelder aus den Tabellen sind standardmäßig automatisch als Schlüsselfelder für die zu erstellenden CursorAdapter markiert. Alle anderen Felder sind standardmäßig als aktualisierbar markiert.

Als Ergebnis erstellt der VFX – CursorAdapter Wizard eine CursorAdapter-Klasse für jede Tabelle aus der ausgewählten Datenbank. Bei jedem CursorAdapter werden die Eigenschaften *CursorSchema*, *Tables*, *SelectCmd*, *KeyFieldList*, *UpdatableFieldList* und *UpdateNameList* vom Wizard eingestellt.

10.4. Datenzugriff mit CursorAdapter

Die Builder von VFX 11.0 unterstützen jetzt die Verwendung von CursorAdaptern in der Datenumgebung. CursorAdapter können in der Datenumgebung genauso wie lokale und remote Ansichten verwendet werden.

CursorAdapter können in allen VFX Buildern und Wizards als Datenquelle angegeben werden. CursorAdapter werden auch als Datenquelle für Auswahllisten unterstützt.

VFX 11.0 enthält eine CursorAdapter-Klasse, die die Grundfunktionalität zum Zugriff auf die Anwendungsdaten enthält. Dies ist die Klasse *CBaseDataAccess* in der Klassenbibliothek *Vfxctrl.vcx* und sollte als Basis für alle CursorAdapter verwendet werden. Diese Klasse stellt sicher, dass die gesamte Anwendung eine gemeinsame Verbindung verwendet und keine überflüssigen Verbindungen geöffnet werden.

10.4.1. Die Klasse CBaseDataAccess

Die neue Klasse *CBaseDataAccess* ermöglicht es basierend auf der VFP-Klasse Cursoradapter auf verschiedene Datenquellen zuzugreifen. Wenn in einer Anwendung der Datenzugriff ausschließlich über die Klasse

CBaseDataAccess erfolgt ist es leicht die Anwendung später auf andere Datenquellen zu portieren. So ist es zum Beispiel einfach möglich zwischen einer VFP-Datenbank und einer SQL Server-Datenbank zu wechseln. Die Datenzugriffseinstellungen für die Klasse *CBaseDataAccess* sind in der Datei *Config.vfx* gespeichert.

Wenn ein Objekt der Klasse *CBaseDataAccess* instanziiert wird, wird aus der Eigenschaft *goProgram.cDataSourceType* der zu verwendende Datenbanktyp gelesen. Wenn der Datenbanktyp *NATIVE* ist, wird eine VFP-Datenbank verwendet. Aus den Eigenschaften *goProgram.cDatadir* und *goProgram.cMainDatabase* werden der Pfad zur Datenbank und der Name der Datenbank gelesen. Bei anderen Datenbanktypen werden die Verbindungsinformationen aus der Methode *GetConnection* des Verbindungs-Managers bezogen.

In der Klassenbibliothek *Appl.vcx* befindet sich die Klasse *CAppDataAccess*, die eine 1:1-Ableitung der Klasse *CBaseDataAccess* ist. Entwickler sollten eigene Erweiterungen oder Änderungen des Datenzugriffs in der Klasse *CAppDataAccess* machen.

Eigenschaften

cConnMgrName – Name des Objekts, das den Namen des Verbindungs-Manager-Objekts enthält. Dieses Verbindungs-Manager-Objekt verwaltet den Datenzugriff der Klasse *CBaseDataAccess*.

cExecuteAfterCursorFill – Der hier eingetragene Befehl wird nach Ausführung der Methode *CursorFill* des CursorAdapters ausgeführt. Hier kann Code eingetragen werden, der die Daten des erstellten Cursors verarbeitet. Mithilfe dieser Eigenschaft kann einem Cursoradapter zur Laufzeit Code hinzugefügt werden.

Filter – Ein logischer Ausdruck mit dem die Daten des erstellten Cursors gefiltert werden.

Order – Der hier angegebene Indexschlüssel wird zur Sortierung des erstellten Cursors verwendet. Indexschlüssel für Cursoradapter können im VFX – Data Environment Builder angelegt werden.

Methoden

CreateIndexes – Der Code dieser Methode wird vom VFX – Data Environment Builder erstellt. Hier werden Befehle zur Erstellung von temporären Indexdateien für den Cursor eingetragen. Diese Methode wird nach Ausführung der Methode *CursorFill()* aufgerufen.

10.5. Datenzugriff bearbeiten mit der Datei Config.vfx

Während der Entwicklung einer Anwendung wird für alle CursorAdapter eine Datenquelle verwendet, die auf dem Entwicklungsrechner zur Verfügung steht. Die Datenquelle auf den Kundenrechnern muss nicht identisch sein. Zum Beispiel kann auf dem Entwicklungsrechner eine SQL Server-Datenbank verwendet werden, während bei den Kunden eine VFP-Datenbank zum Einsatz kommt.

Auch wenn auf dem Entwicklungsrechner und auf dem Kundenrechner eine SQL Server-Datenbank verwendet werden soll, so wird der Name des SQL Servers auf beiden Rechnern unterschiedlich sein. Daher ist in der Regel auf dem Entwicklungsrechner und dem Kundenrechner eine andere Verbindungszeichenfolge erforderlich.

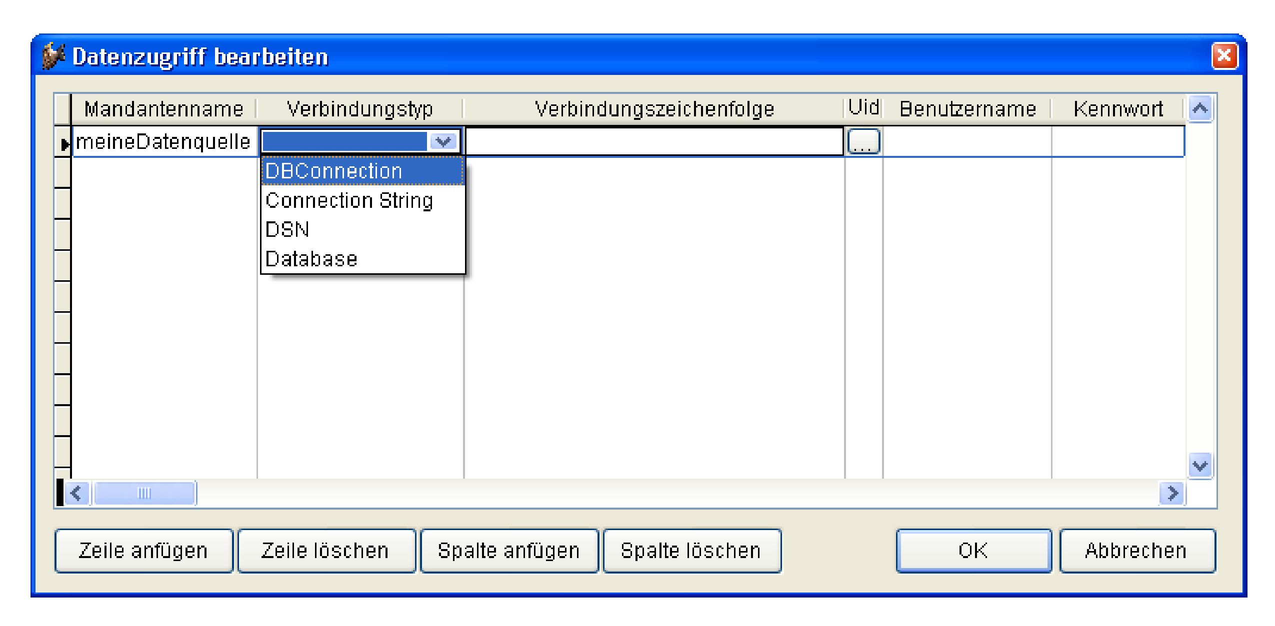
VFX verwendet einen eigenen Verbindungs-Manager um eine Verbindung zur Datenquelle herzustellen. Dieser Verbindungs-Manager wird als Child-Objekt des Anwendungsobjekts instanziiert und steht über die Referenz *goProgram.oConnMgr* zur Laufzeit zur Verfügung.

CursorAdapter-Objekte basierend auf der Klasse *CBaseDataAccess* verwenden das Objekt *goProgram.oConnMgr*, eine Instanz der Klasse *CConnectionMgr*, um eine Verbindung zur Datenquelle herzustellen. Die Einstellungen für das *goProgram.oConnMgr* Objekt werden aus der Datei *Config.vfx* gelesen. In dieser Datei befinden sich die Informationen über die von der Anwendung verwendete Datenquelle.

Die Datei *Config.vfx* enthält aus Sicherheitsgründen verschlüsselte Daten, die zur Verbindung mit der Kundendatenbank verwendet werden, zum Beispiel Typ der Datenquelle, Verbindungszeichenfolge und andere. Das Kennwort zur Verschlüsselung ist in der Eigenschaft *goProgram.cConfigPassword* gespeichert. VFX-Entwickler sollten dieses Kennwort selbst zuweisen.

Die Datei *Config.vfx* kann vom Entwickler erstellt und zusammen mit der Anwendung ausgeliefert werden. Wenn beim Start der Anwendung keine Datei *Config.vfx* gefunden wird, verwendet die VFX-Anwendung die Datenbank, die in der Eigenschaft *goProgram.cDataDir* hinterlegt ist. Wenn *goProgram.cDataDir* eine leere Zeichenkette zugewiesen ist, werden die Datenbankinformationen aus der Tabelle *Vfxpath.dbf* gelesen.

Benutzer mit Administratorrechten können die Datei *Config.vfx* später über den Menüpunkt *Extras, Datenzugriff bearbeiten* bearbeiten.



Für jeden Kunden kann gewählt werden, ob mit einer VFP-Datenbank oder einer Remote-Datenbank gearbeitet werden soll. Die Datei *Config.vfx* kann auch mehrere Datensätze enthalten. Wenn mehr als ein Datensatz vorhanden ist, erscheint beim Start der Anwendung ein Datenbankauswahldialog.

Es kann eine Verbindung aus einer VFP-Datenbank verwendet werden. Zur Laufzeit wird der Name der Verbindung in der Eigenschaft *cDBCConn* des Objekts *goProgram* gespeichert. In der Datei *Config.vfx* wird der Name der zu verwendenden Datenbank gespeichert. Beim Start der Anwendung werden die Informationen zur Datenbank aus dieser Datei gelesen.

Um eine ODBC-Verbindung zu benutzen, kann eine Verbindungszeichenfolge oder eine existierende DSN verwendet werden. Wenn eine Verbindungszeichenfolge als Datenquelle gewählt wird, kann über die Schaltfläche ein Dialog angezeigt werden, der hilft eine gültige Verbindungszeichenfolge zu erstellen.

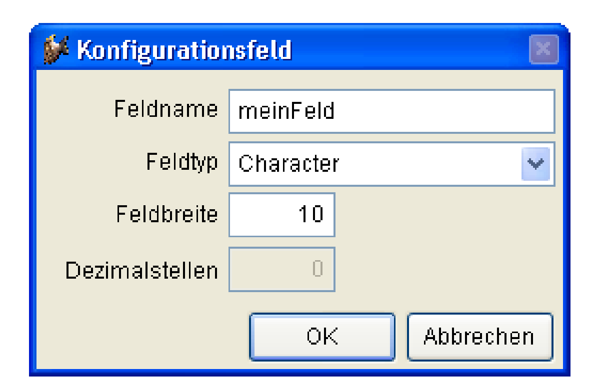
Wenn eine DSN als Datenquelle gewählt wird, können ein Benutzername und ein Kennwort eingegeben werden, die zur Anmeldung bei der Datenquelle zur Laufzeit verwendet werden. Wenn hier kein Benutzername und Kennwort eingegeben werden und die Datenquelle eine Anmeldung erfordert, erscheint zur Laufzeit ein Anmeldedialog, der den Anwender zur Eingabe von Benutzername und Kennwort auffordert.

Die Datei *Config.vfx* entspricht in etwa der Datei *Vfxpath.dbf*, die wir aus früheren VFX-Versionen kennen. Alle in der Tabelle *Vfxpath.dbf* vorhandenen Felder sind auch in *Config.vfx* vorhanden.

Es können mehrere Zeilen vorhanden sein, die auf verschiedene Typen von Datenquellen zugreifen. So kann ein Kunde mit einer Anwendung beim Programmstart entscheiden, ob er auf einer VFP-Datenbank oder auf verschiedenen Server-Datenbanken arbeiten will.

Durch die Verschlüsselung der Datei *Config.vfx* ist eine in VFP-Anwendungen bisher nicht erreichte Sicherheit erreicht worden.

Genau wie bei der Tabelle *Vfxpath.dbf* können der Datei *Config.vfx* eigene Felder hinzugefügt werden, deren Werte dann zur Laufzeit der Anwendung zur Verfügung stehen. Die Schaltfläche *Add Column* zeigt einen Dialog an, in dem Name und Typ von neuen Feldern eingegeben werden können.



10.6. Wechsel zwischen DBC und SQL Server

Wenn eine VFX 11.0-Anwendung so konstruiert ist, dass der Datenzugriff ausschließlich über CursorAdapter erfolgt, ist der Wechsel zwischen einem DBC und einer SQL Server-Datenbank nachträglich problemlos möglich.

Nehmen wir an, wir haben eine Anwendung mit einem DBC als Datenquelle entwickelt. Bei der Entwicklung haben wir darauf geachtet, dass jeglicher Datenzugriff nur über CursorAdapter erfolgt. Jetzt möchte ein Kunde diese Anwendung mit einer SQL Server-Datenbank laufen lassen.

Dafür muss die VFP-Datenbank zunächst auf SQL Server portiert werden. Das können wir mit dem Upsizing-Assistenten aus VFP machen, aber auch andere Werkzeuge, wie zum Beispiel xCase sind für diese Aufgabe geeignet.

Für den Zugriff auf die SQL Server-Datenbank kann eine DSN eingerichtet werden. Dies stellt aber auch wieder ein Sicherheitsrisiko dar, weil eine DSN manipuliert werden kann. Sicherer ist es in der Datei *Config.vfx* eine Verbindungszeichenfolge für den Datenzugriff zu wählen. Dadurch ist man unabhängig von weiteren Einstellungen auf Betriebssystemebene und hat alle Informationen über den Datenzugriff innerhalb der Anwendung gespeichert.

Die SQL Server-Datenbank wird auf dem Server des Kunden installiert. Die fertige Anwendung wird mit einer leeren Datei *Config.vfx* ausgeliefert. Dadurch erscheint beim Start der Anwendung beim Kunden automatisch der Dialog zur Bearbeitung der Datenquellen. Die Verbindung zum beim Kunden installierten SQL Server kann mit Benutzername und Kennwort eingegeben werden und es kann mit der Anwendung gearbeitet werden.

10.7. Formulare basierend auf Ansichten

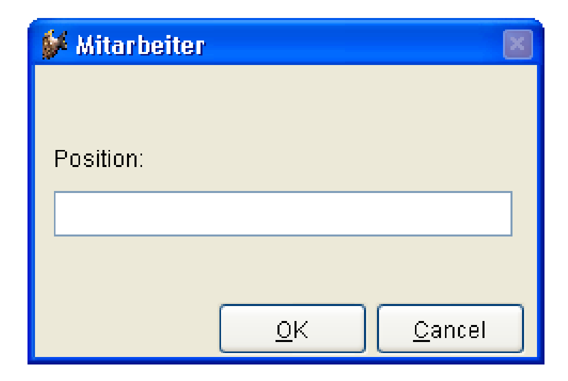
Bei der Entwicklung von VFX wurde großer Wert darauf gelegt, dass sowohl direkt mit VFP-Tabellen, als auch mit lokalen Ansichten und mit Remote Ansichten gearbeitet werden kann. Ansichten können insbesondere keine Indexschlüssel haben. VFX muss also in jedem Fall, in dem eine Sortierung benötigt wird, eine temporäre Indexdatei erstellen.

Ansichten können für jeden VFX-Formulartyp als Datenquelle verwendet werden. Es ist möglich OneToMany-Formulare oder Parent/Child-Konstruktionen auf Ansichten basieren zu lassen. Auch ist die Verwendung von Ansichten bei Auswahllisten möglich. Eine VFX-Anwendung kann somit als Frontend z. B. für einen SQL-Server oder andere Remote-Datenquellen verwendet werden.

In den meisten Fällen sind Ansichten parametrisiert. Die Parameter müssen vor Abfrage der Daten der Ansicht bekannt sein. Zur Eingabe der Ansichtsparameter stellt VFX die Formularklasse *CAskViewArg* zur Verfügung. Das Datenbearbeitungsformular wird wie gewohnt mit dem VFX – Form Builder erstellt. Bei der Ansicht in der Datenumgebung wird die Eigenschaft *nodataonload* auf .T. gesetzt. Das bedeutet, dass die Ansicht beim Laden des Formulars geöffnet wird, ohne dass Daten abgefragt werden.

Jetzt wird ein neues Formular basierend auf der Klasse *CAskViewArg* erstellt. Die Steuerelemente, die als Controlsource Felder enthalten, die auch als Ansichtsparameter verwendet werden, können über die Zwischen-

ablage vom Bearbeitungsformular auf das Formular basierend auf der Klasse *CAskViewArg* kopiert werden. In der Eigenschaft *cviewparameter* ist der Name des Ansichtsparameters einzutragen. Den Steuerelementen können geeignete Bezeichnungen hinzugefügt werden. Das Formular ist damit fertig und kann gespeichert werden.



Aus dem Bearbeitungsformular muss nun noch das Formular basierend auf der Klasse *CAskViewArg* aufgerufen werden. Dies geschieht am Ende des *Init()*-Ereignis:

```
do form <Formular zur Eingabe der Ansichtsparameter> with this
```

Es ist auch möglich zur Laufzeit des Formulars das Formular zur Eingabe der Ansichtsparameter erneut aufzurufen. Wenn der Aufruf aus einem Steuerelement, zum Beispiel aus dem *Click()*-Ereignis einer Schaltfläche erfolgt, muss der Aufruf so aussehen:

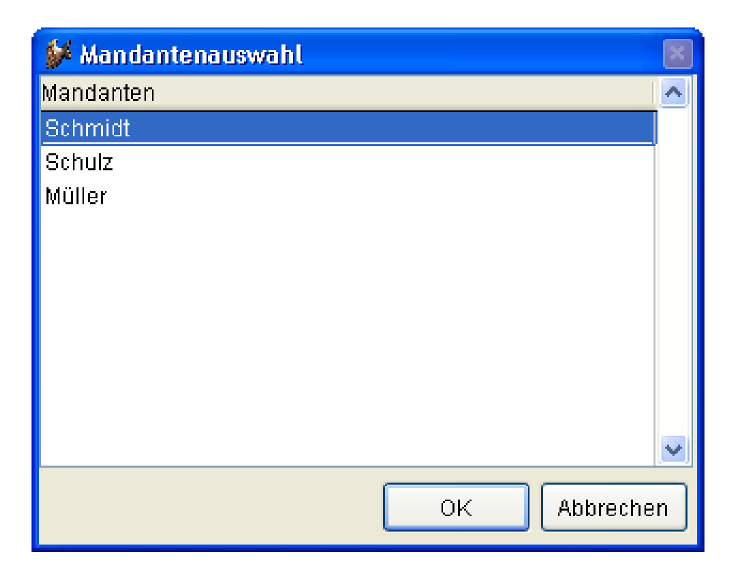
```
do form <Formular zur Eingabe der Ansichtsparameter> with thisform
```

Mehr ist bei der Arbeit mit Ansichten nicht zu beachten. Alles Weitere erledigt VFX.

10.8. Multi-Client-Support

Standardmäßig arbeitet eine VFX-Anwendung mit genau einer Datenbank, so wie es im VFX – Application Wizard eingetragen wurde. Auf Wunsch kann eine Mandantenfähigkeit eingebaut werden. Dazu ist die Eigenschaft *cdatadir* der Anwendungsklasse *CFoxAppl* in *Appl.vcx* auf einen Leerstring zu setzen.

Wenn die Datei *Config.vfx* zur Laufzeit gefunden wird, werden die Datenzugriffsinformationen aus dieser Datei benutzt. Die Verwendung der *Datei Config.vfx* ist oben im Kapitel *Datenzugriff bearbeiten mit der Datei Config.vfx* beschrieben. Wenn beim Start der Anwendung keine Datei *Config.vfx* gefunden wird, verwendet die VFX-Anwendung die Datenbank, die in der Eigenschaft *goProgram.cDataDir* hinterlegt ist. Wenn *goProgram.cDataDir* eine leere Zeichenkette zugewiesen ist, werden die Datenbankinformationen aus der Tabelle *Vfxpath.dbf* gelesen. Diese Tabelle muss sich im gleichen Ordner wie die ausführbare Programmdatei befinden. Wenn in dieser Tabelle genau ein Datensatz enthalten ist, wird der dort eingetragene Datenpfad verwendet. Enthält die Tabelle mehr als einen Datensatz erscheint beim Start der Anwendung ein Dialog zur Auswahl der gewünschten Datenbank.



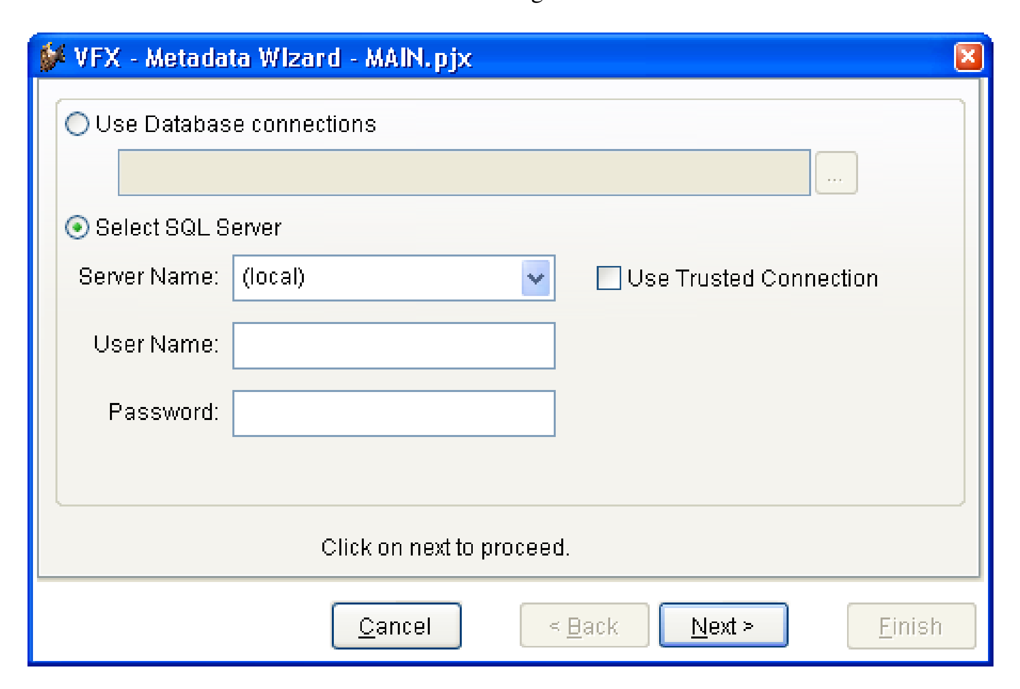
10.9. Aktualisierung der Kundendatenbank

10.9.1. Verwendung von VFP-Datenbanken

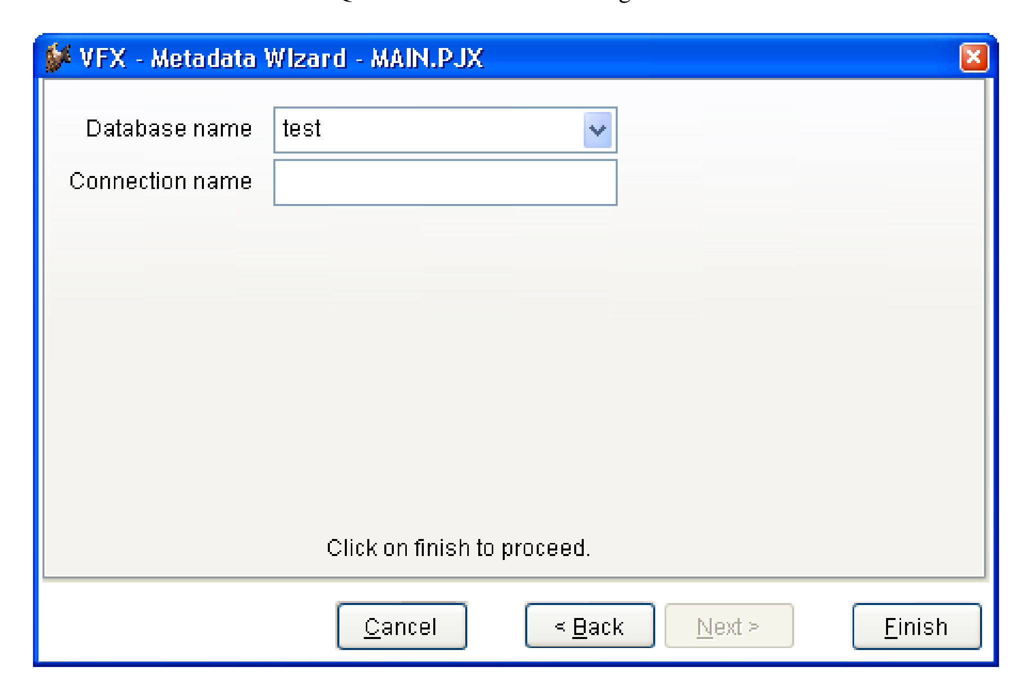
VFX enthält Routinen um eine Aktualisierung der Datenbank beim Kunden automatisch durchzuführen. Dazu wird unterhalb des Datenordners ein Ordner mit dem Namen *Update* angelegt. In diesen Ordner wird die Datenbank mit allen Tabellen, jedoch ohne Daten, kopiert. Es können so auch freie Tabellen aktualisiert werden. Beim Programmstart wird die Datenbank im Datenordner aktualisiert. Es können der Datenbank auf diese Weise neue Tabellen, neue Felder in Tabellen, neue Indexschlüssel und neue Ansichten hinzugefügt werden. Ebenso werden nicht mehr benötigte Tabellen, Felder usw. gelöscht. Anschließend werden alle Dateien im *Update*-Ordner gelöscht. Mit dieser Methode können auch freie Tabellen aktualisiert werden.

10.9.2. Verwendung von SQL Server-Datenbanken

Der VFX – Metadata Wizard hilft Ihnen Metadaten aus Ihrer aktuell benutzten SQL Server-Datenbank zu erstellen. Die Metadaten können zur Aktualisierung der Datenbank beim Kunden verwendet werden.



Wahlweise kann die Verbindung aus einer VFP-Datenbank ausgelesen werden um die Verbindung zu einem SQL Server herzustellen oder der SQL Server kann manuell ausgewählt werden.



Der Metadata Wizard erstellt die Tabelle *Datadict.dbf*. Dies ist eine freie Tabelle, in der die Struktur der SQL Server Datenbank inklusiv Constraints, benutzerdefinierten Datentypen, Regeln, Ansichten und gespeicherten Prozeduren gespeichert wird. Der Metadata Wizard durchsucht das aktive Projekt nach Verbindungen und analysiert die Struktur der Datenbank. Wenn die Tabelle *Datadict.dbf* an die Kunden weitergegeben wird, wird die Struktur der dortigen Datenbank aktualisiert. Dabei wird wieder die bestehende Verbindung zum Zugriff auf die Datenbank verwendet.

10.10. Indexdateien

VFX macht von vorhandenen Indexschlüsseln bestmöglichen Gebrauch. Für die inkrementelle Suche in VFX-Power Grids durchsucht VFX automatisch alle vorhandenen Indexschlüssel der verwendeten Tabelle. Für Zeichenfelder wird ein Indexschlüssel mit *UPPER()*-Klausel erwartet. Für Datumsfelder wird ein Indexschlüssel mit *DTOS()*-Klausel erwartet.

Wenn VFX keinen passenden Indexschlüssel findet, wird eine temporäre Indexdatei angelegt. Diese Indexdatei wird gelöscht, sobald das Formular geschlossen wird. Ferner wird die Indexdatei gelöscht, wenn das Formular in den Bearbeitungsmodus oder in den Einfügemodus wechselt sowie beim Löschen von Datensätzen. Das ist sinnvoll, weil laufende Transaktionen, wie sie zum Beispiel im RI-Code verwendet werden, zu VFP-Laufzeitfehlern führen würden, wenn temporäre Indexdateien geöffnet sind. VFP erlaubt keine temporären Indexdateien, wenn mit Transaktionen gearbeitet wird.

Wenn in einem Formular Transaktionen verwendet werden, kann auf Wunsch nach der Datenbearbeitung der zuvor gültige Indexschlüssel wieder erstellt werden. Dem Anwender wird vorgetäuscht, dass die gewählte Sortierfolge ständig erhalten bleibt. Stellen Sie dafür im VFX – Application Builder *Recreate temporary index files after editing* ein.

Wenn in einem Formular und jeglichem daraus aufgerufenen Code keine Transaktionen ausgeführt werden, also in den beteiligten Tabellen auch kein RI-Code hinterlegt ist, können Sie VFX – Application Builder einstellen, dass temporäre Indexdateien bei der Datenbearbeitung nicht gelöscht werden sollen. Markieren Sie hierfür die Felder *Disable clearing indexes when editing data*, *Disable clearing indexes when inserting records* bzw. *Disable clearing indexes when deleting records*.

Temporäre Indexdateien werden in jedem Fall beim Schließen eines Formulars gelöscht.

11. Anwendungsschutz durch Produktaktivierung

Das Ziel der Produktaktivierung ist die unerlaubte Verwendung der Anwendung auf nicht aktivierten Computern zu verhindern.

Der Anwendungsschutz durch Produktaktivierung kann im VFX – Application Wizard auf der Seite 3. Options durch aktivieren des Kontrollkästchens *Enable product activation* für ein neu zu erstellendes Projekt eingeschaltet werden.

Später kann diese Einstellung mithilfe des VFX – Application Builder geändert werden. Die Eigenschaft *goProgram.lUseActivation* muss auf .T. gesetzt werden, um die Produktaktivierung einzuschalten. Wenn die Eigenschaft *goProgram.lUseActivation* auf .F. gesetzt ist, ist die Anwendung nicht durch die Produktaktivierung geschützt.

Zu jeder Anwendung können bis zu 32 Rechte vergeben werden. Jedes Recht kann unabhängig von den anderen Rechten aktiviert werden.

11.1. Liste der verwendeten Begriffe

Systemspezifischer Wert – Ein systemspezifischer Wert, zum Beispiel die Seriennummer einer Hardware-Komponente oder das Erstellungsdatum einer bestimmten Datei oder ein Schlüssel aus der Windows-Registrierungsdatenbank. Die zu verwendete Datei und der zu verwendende Schlüssel aus der Windows-Registrierungsdatenbank können vom Entwickler festgelegt werden.

Aktivierungsregel – Für jede Anwendung kann eine eindeutige Aktivierungsregel angelegt werden. Diese Regel setzt sich aus einer Reihe systemspezifischer Werte zusammen, die einen PC eindeutig identifizieren. Bei der Erstellung der Aktivierungsregel können Textbearbeitungsfunktionen verwendet werden.

Installationsschlüssel – Dies ist eine Zeichenkette, die Informationen über die im PC des Anwenders eingesetzte Hardware enthält. Der Installationsschlüssel wird vom Entwickler benötigt um einen Aktivierungsschlüssel erstellen zu können.

Aktivierungsschlüssel – Dies ist eine Zeichenkette, die die Berechtigungen für einen speziellen PC enthält. Der Aktivierungsschlüssel wird vom Entwickler anhand des Installationsschlüssels erstellt. Der Aktivierungsschlüssel ist für andere PCs nutzlos.

Installationsdatum – An diesem Datum wurde eine Anwendung erstmalig auf einem PC gestartet.

11.2. Das Funktionsprinzip

Wenn der Anwendungsschutz durch Produktaktivierung aktiviert ist, wird beim Start der Anwendung das Objekt *goProgram.SecurityRights* instanziiert. Dieses Objekt hat Eigenschaften mit den Namen der Benutzerrechte, die der Entwickler definiert hat. Jede dieser Eigenschaften kann einen von drei Werten annehmen:

-1 – Die Anwendung ist nicht aktiviert. In diesem Fall kann der Entwickler entscheiden welche Aktion ausgeführt werden soll. Der Anwender könnte zum Beispiel begrenzten Zugriff auf Funktionen haben, solange die Anwendung nicht aktiviert ist.

0 – Die Anwendung ist aktiviert, aber der Anwender hat nicht das Recht diese Aktion auszuführen.

1 – Die Anwendung ist aktiviert und der Anwender hat das Recht die Aktion auszuführen.

Wenn der Anwendungsschutz durch Produktaktivierung aktiviert ist, werden der Aktivierungsschlüssel und das Datum des ersten Starts der Anwendung in einer Ini-Datei gespeichert. Der Entwickler kann den Namen dieser Ini-Datei selbst wählen, sodass jede Anwendung ihre eigene Ini-Datei verwendet. Der Standardname ist *VFX.ini*. Die Ini-Datei wird im Windows-Ordner gespeichert.

Der Aktivierungsschlüssel wird durch die Aktivierungsregel verschlüsselt. Der Schutz kann durch Hinzufügen von Zeichenkonstanten, Schlüsseln aus der Windows-Registrierungsdatenbank und durch das Erstellungsdatum

einer beliebigen Datei weiter verbessert werden. Diese Kombination kann für jede Anwendung getrennt festgelegt werden, sodass jede Anwendung ihre eigenen Aktivierungsregeln hat.

Zusätzlich zu diesen Einstellungen kann der Entwickler den Typ des Schutzes festlegen.

Der Standardschutz erstellt die Ini-Datei beim ersten Start der Anwendung. Das während des Erstellens der Ini-Datei aktuelle Systemdatum wird in der Datei gespeichert. Dieses Datum steht während der Ausführung der Anwendung in der Eigenschaft *goProgram.InstallationDate* zur Verfügung und kann dazu verwendet werden die Laufzeit der Anwendung zu beschränken. Der Nachteil dieses Schutzes ist, dass der Anwender die erstellte Ini-Datei löschen kann und die Ini-Datei beim nächsten Start der Anwendung mit einem neuen Datum erneut erstellt wird.

Um eine solche Manipulation durch den Anwender auszuschließen, kann der Entwickler einen erweiterten Schutz einstellen. Hierbei wird eine zusätzliche Datei verwendet, die mit der Anwendung vertrieben werden muss. Der Standardname dieser Datei heißt *FirstInstall.txt*. Der Dateiname kann mit der Eigenschaft *cFirstInstall* aus der Klasse *CActivation (Appl.vcx)* eingestellt werden. Die Datei *FirstInstall.txt* wird im Windows-Ordner abgelegt.

Wenn der Entwickler den Schutz mit der Datei *FirstInstall.txt* auswählt, wird sich die Anwendung folgendermaßen verhalten. Beim Start der Anwendung wird zunächst die Ini-Datei überprüft. Wenn diese Datei existiert, wird das Datum des ersten Starts der Eigenschaft *goProgram.InstallationDate* zugewiesen und die Benutzerrechte werden entsprechend dem Aktivierungsschlüssel eingestellt.

Wenn die Ini-Datei nicht existiert wird angenommen, dass dies der erste Start der Anwendung ist. Wenn dies der Fall ist, wird zusätzlich überprüft, ob die Datei *FirstInstall.txt* existiert. Wenn diese Datei existiert ist sichergestellt, dass die Anwendung wirklich zum ersten Mal gestartet wird. Das Systemdatum wird jetzt in der Ini-Datei gespeichert und die Datei *FirstInstall.txt* wird gelöscht. Wenn ein Anwender nun versucht eine Anwendung zu reaktivieren indem er die Ini-Datei löscht, wird die Ausführung der Anwendung beendet, weil die Datei *FirstInstall.txt* nicht existiert. Dieser erweiterte Schutz der Anwendung bedeutet eine bessere Sicherheit. Der Entwickler darf jedoch nicht vergessen die Datei *FirstInstall.txt* beim Vertrieb der Anwendung mit auszuliefern.

Wenn der Anwender die installierte Anwendung aktivieren möchte, muss er seinen Installationsschlüssel an den Entwickler senden. Der Installationsschlüssel kann auf drei verschiedene Arten an den Entwickler gesendet werden. Die gewünschte Art kann in der Eigenschaft *nRegWay* eingestellt werden:

> *0* – Der Installationsschlüssel wird in einem Dialog angezeigt. Der Anwender kann den Schlüssel kopieren und in einer anderen Anwendung (zum Beispiel in einer E-Mail) einfügen.
>
> *1* – Der Installationsschlüssel wird in einer Datei gespeichert. Diese Datei kann später an den Entwickler gesendet werden. Der Dateiname wird in der Eigenschaft *cParamFile* hinterlegt.
>
> *2* – Der Installationsschlüssel wird in einer Datei gespeichert und sofort als E-Mail-Anhang an den Entwickler geschickt. Der Dateiname muss in der Eigenschaft *cParamFile* hinterlegt werden. Die E-Mail-Adresse des Entwicklers muss in der Eigenschaft *cRegEMail* eingetragen werden.
>
> *11* – Nach Anzeige des Registrierungsdialogs wird der Installationsschlüssel in einer Datei gespeichert. Diese Datei kann später an den Entwickler gesendet werden. Der Dateiname wird in der Eigenschaft *cParamFile* hinterlegt.
>
> *12* – Nach Anzeige des Registrierungsdialogs wird der Installationsschlüssel in einer Datei gespeichert und sofort als E-Mail-Anhang an den Entwickler geschickt. Der Dateiname muss in der Eigenschaft *cParamFile* hinterlegt werden. Die E-Mail-Adresse des Entwicklers muss in der Eigenschaft *cRegEMail* eingetragen werden.

Der Installationsschlüssel hat einen numerischen Wert mit 10 Stellen Länge. Der Anwender könnte den Installationsschlüssel per E-Mail an den Entwickler senden oder auf einer Registrierungs-Website eintragen. Über den VFX-Menüpunkt *Activation, Customer List* wird die VFX-Kundenverwaltung geöffnet. Der Entwickler trägt den Installationsschlüssel im „Create Activation Key"-Assistenten ein um einen Aktivierungsschlüssel für den Anwender zu erstellen. Der generierte Aktivierungsschlüssel wird dann an den Anwender geschickt und vom

Anwender im Aktivierungsformular eingegeben um die Anwendung zu aktivieren. Wahlweise kann die Datei mit dem Aktivierungsschlüssel auch einfach im Ordner der Exe-Datei gespeichert werden. Beim nächsten Start der Anwendung wird der Aktivierungsschlüssel aus dieser Datei gelesen.

Die Aktivierungsinformationen werden auf dem PC des Kunden in einer Ini-Datei gespeichert. Der Name dieser Ini-Datei wird in der Eigenschaft *cINIFileName* der Klasse *CVFXAcvtivation (Appl.vcx)* eingetragen. Der Standardwert ist *VFX.ini*.

Der Entwickler kann wählen, ob die einfache Produktaktivierung verwendet werden soll oder ob zusätzlich die Datei *FirstInstall.txt* benutzt werden soll, um den ersten Start der Anwendung zu protokollieren. Der Name dieser Datei kann in der Eigenschaft *cFirstInstall* der Klasse *CVFXAcvtivation (Appl.vcx)* eingetragen werden. Der Standardwert ist *FirstInstall.ini*.

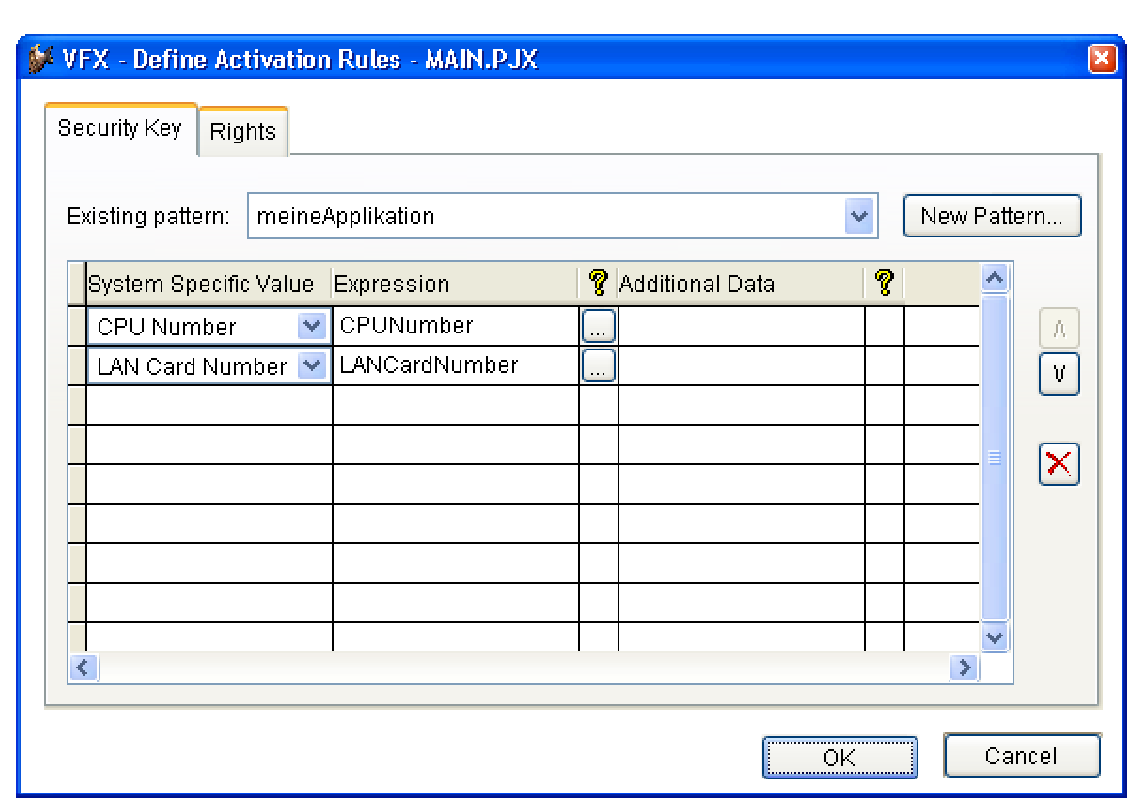
Wenn die Datei *FirstInstall.txt* verwendet werden soll, muss diese Datei mit der Anwendung vertrieben werden. Das Installationsprogramm muss diese Datei im Windows-Ordner speichern. Das Aktivierungsobjekt wird diese Datei beim ersten Start der Anwendung löschen. In diesem Moment wird das Installationsdatum in der Ini-Datei gespeichert. Später wird bei jedem Start der Anwendung in der Ini-Datei geprüft, ob das Installationsdatum vorhanden ist. Wenn das Datum fehlt und wenn die Datei *FirstInstall.txt* nicht vorhanden ist, wird davon ausgegangen, dass an der Installation manipuliert wurde und die Ausführung der Anwendung wird beendet.

Wenn die Datei *FirstInstall.txt* nicht verwendet wird, wird die Ini-Datei neu erstellt, falls sie nicht vorhanden ist.

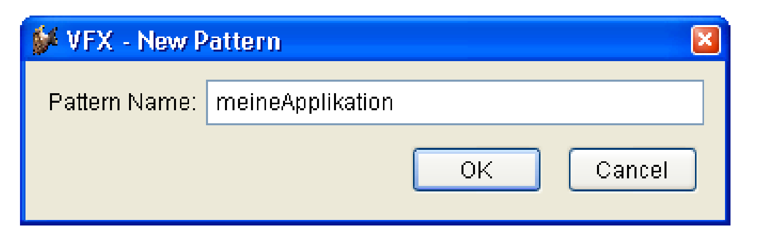
Das Installationsdatum kann auf zwei Arten ermittelt werden. Entweder wird das Systemdatum verwendet oder es wird das Erstellungsdatum einer bestimmten Datei verwendet. Wenn das Erstellungsdatum einer Datei verwendet werden soll, muss der Name dieser Datei in der Eigenschaft *cRegFileName* der Klasse *CVFXActivation* gespeichert werden.

11.3. Die Definition der Aktivierungsregeln

Starten Sie den Dialog VFX – Define Activation Rules über den VFX-Menüpunkt *Activation, Define Activation Rules*.



Wenn der „Define Activation Rules"-Assistent das erste Mal für ein Projekt gestartet wird, muss eine neue Regel für dieses Projekt angelegt werden.



Auf der Seite Security Key des Assistenten befindet sich eine Combobox aus der eine Regel für das aktuelle Projekt ausgewählt werden kann. In dem darunter liegenden Grid können so viele Zeilen hinzugefügt werden, wie benötigt werden. Aus allen Zeilen des Grids wird in ein Schlüssel generiert, der in der Eigenschaft *cactpattern* der Klasse *CVfxactivation* gespeichert wird. Die Anwendung beim Kunden erkennt anhand dieses Schlüssels welche systemspezifischen Werte des PCs zur Generierung des Installationsschlüssels verwendet werden müssen. Der Installationsschlüssel stellt sicher, dass die Anwendung nur auf dem Computer ausgeführt wird, für den der Aktivierungsschlüssel erstellt wurde.

In der ersten Spalte des Grid kann ein systemspezifischer Wert ausgewählt werden. In einer Combobox sind hier alle möglichen Parameter aufgeführt, die zur Erstellung des Installationsschlüssels verwendet werden können. Zusätzlich können Zeichenkettenfunktionen angewendet werden, um den Wert zu verändern.

Zum Beispiel sollen anstelle der vollständigen Seriennummer einer Festplatte nur die letzten vier Stellen zur Erstellung des Installationsschlüssels verwendet werden. Aus der Combobox in der ersten Spalte wird „HDD Factory Serial Number" ausgewählt. Die VFX-Systemvariable, die diesem Parameter entspricht heißt

„HDDFactoryNumber“ und erscheint in der zweiten Spalte. Um nur die letzten vier Stellen zu verwenden, muss der folgende Ausdruck in der zweiten Spalte eingetragen werden: RIGHT(ALLTRIM(HDDFactoryNumber),4).

Wenn einer der systemspezifischen Werte „File Creation Date“ oder „Registry Key Value“ verwendet werden soll, müssen weitere Parameter angegeben werden. Wenn das Erstellungsdatum einer Datei verwendet werden soll, muss der Name der Datei angegeben werden. Um einen Windows-Registrierungsschlüssel verwenden zu können, muss die Bezeichnung des Schlüssels eingegeben werden. Dies geschieht in der Spalte „Additional Data“.

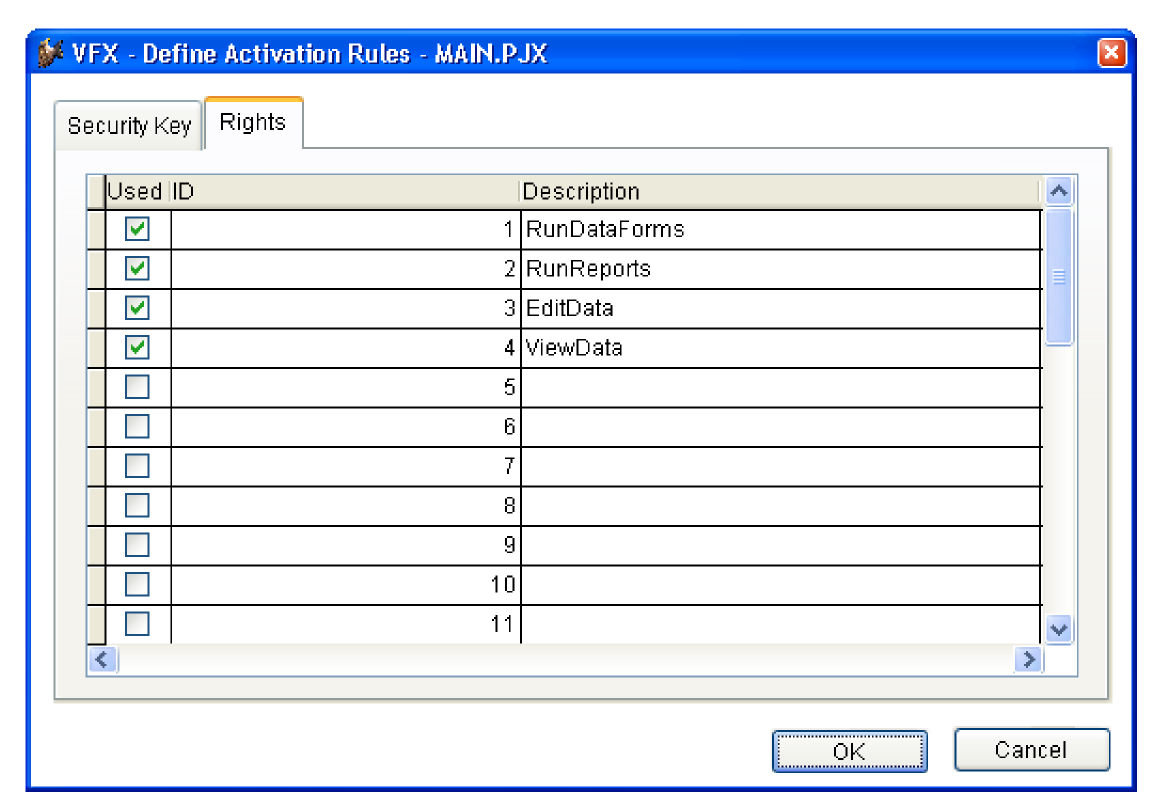
Aus den Aktivierungsregeln wird auf dem PC des Anwenders ein Installationsschlüssel erstellt. Dabei werden alle in den Aktivierungsregeln enthaltenen Parameter berücksichtigt. Wenn nur ein Parameter auf dem PC des Anwenders verändert wird, wird die Installation ungültig und der Anwender muss einen neuen Aktivierungs-schlüssel anfordern, entsprechend seiner geänderten Hardware.

Es können so viele Zeilen dem Grid hinzugefügt werden, wie benötigt werden. Die Zeilen im Grid können mit den Pfeiltasten am rechten Rand des Assistenten in eine andere Reihenfolge gebracht werden. Durch verschieben der Zeilen im Grid ändern sich die Aktivierungsregeln.

Nach der Definition der Aktivierungsregeln wird das Muster in der Eigenschaft *cActPattern* der Klasse *CVFXActivation (Appl.vcx)* gespeichert.

ACHTUNG: Der Wert der Eigenschaft *cActPattern* darf niemals gelöscht werden! Ohne diesen Wert ist es nicht möglich Aktivierungsschlüssel zu erstellen!

Auf der Seite *Rights* können bis zu 32 verschiedene Benutzerrechte angelegt werden. Damit kann der Zugriff auf bis zu 32 Module einer Anwendung gesteuert werden. Beispielsweise können Rechte angelegt werden, die es dem Anwender erlauben Formulare zu starten (*RunDataForms*), Berichte zu drucken (*RunReports*), Daten zu bearbeiten (*EditData*), Daten anzusehen (*ViewData*) usw. Zur Laufzeit der Anwendung können die einzelnen Berechtigungen überprüft werden und ggf. wird die entsprechende Aktion ausgeführt.



Alle Benutzerrechte stehen zur Laufzeit als Eigenschaften des global sichtbaren Objekts *goProgram.SecurityRights* zur Verfügung, sodass an jeder Stelle der Anwendung darauf zugegriffen werden kann.

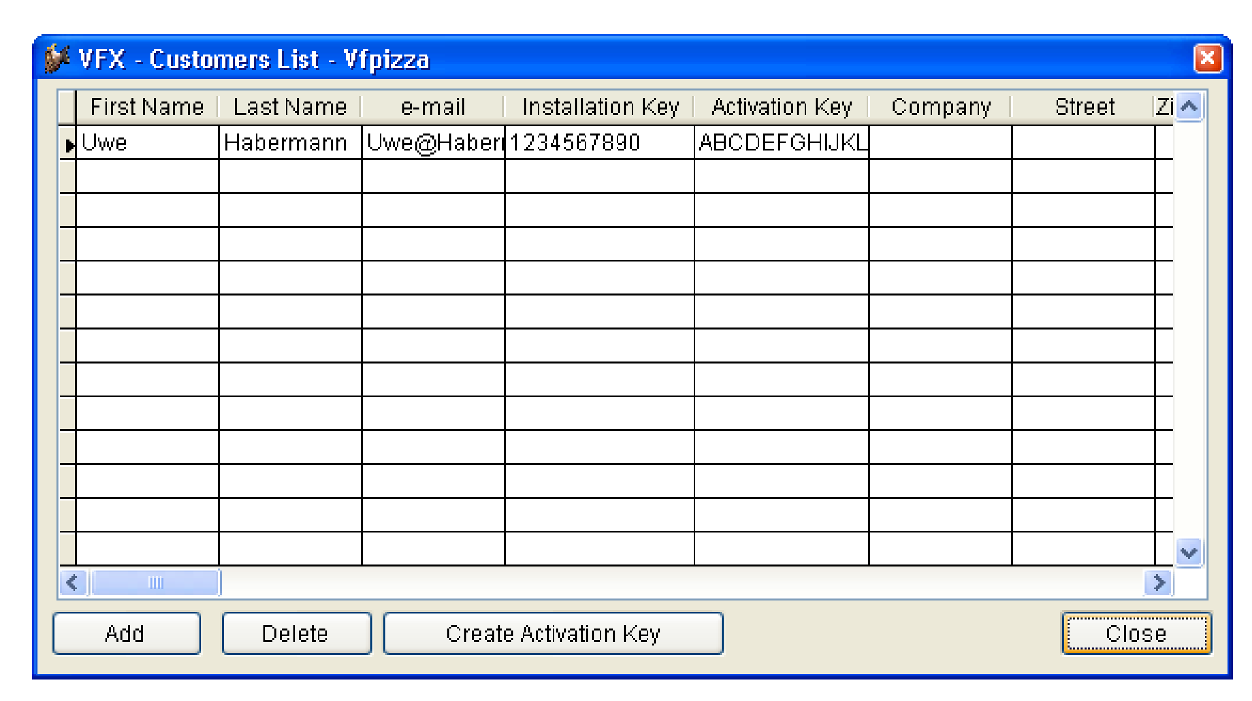
Wenn die Anwendung nicht aktiviert ist, haben alle Benutzerrechte den Wert -1. Wenn die Anwendung aktiviert ist, hat ein Benutzerrecht den Wert 1, wenn die Aktion erlaubt ist und 0, wenn die Aktion nicht erlaubt ist.

Um im Assistenten ein Recht einzutragen muss zuerst das Kontrollkästchen in der ersten Spalte markiert werden. Dann wird ein Name für das Recht eingetragen. Zur Laufzeit der Anwendung wird eine Eigenschaft des SecurityRights-Objekts mit diesem Namen angelegt. Daher müssen bei der Eingabe des Namens die Konventionen zur Namensgebung von VFP beachtet werden!

ANMERKUNG: Anwendungsrechte sind für jede Anwendung unterschiedlich. Die Rechte, die für eine andere Anwendung erstellt wurden, können nicht verwendet werden. Auch wenn ähnliche Rechte benötigt werden, müssen diese neu erstellt werden. Die Anwendungsrechte werden in der Tabelle *Vfxapprights.dbf* im Projektordner gespeichert.

11.4. Erstellen eines Aktivierungsschlüssels

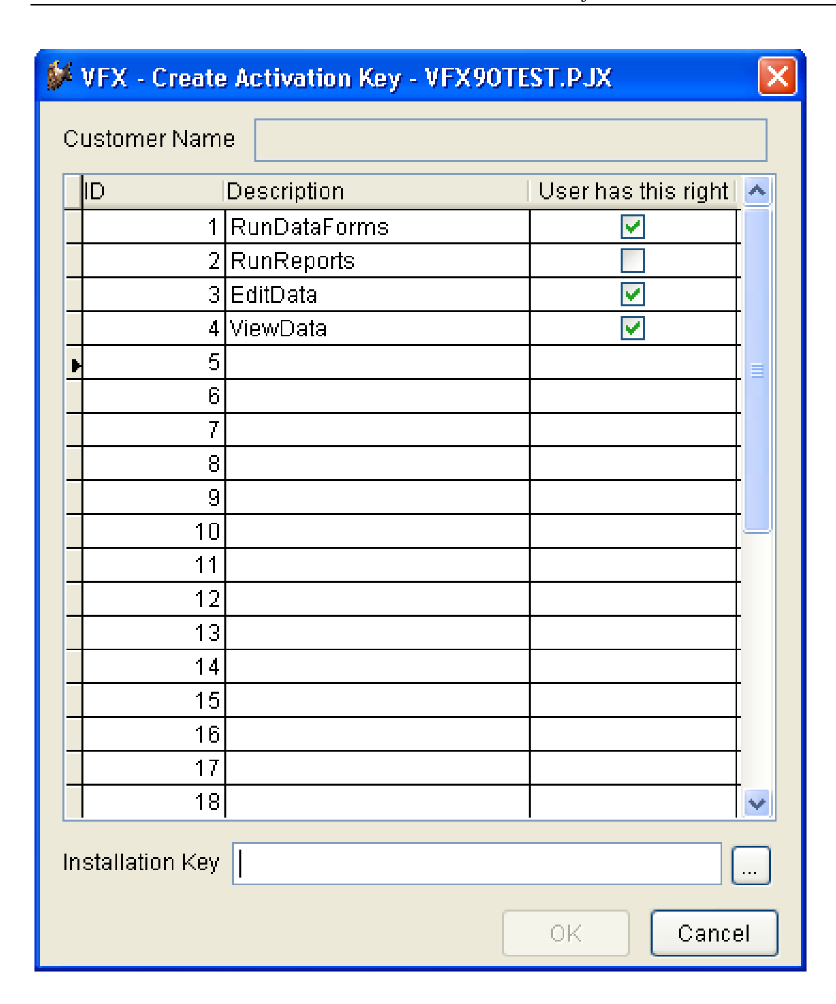
VFX 10.0-Anwendungen können vor unbefugter Nutzung mit einem Aktivierungsschlüssel geschützt werden. Die Daten der Kunden, die einen Aktivierungsschlüssel erhalten haben, können mit umfangreichen Benutzerdaten und Benutzerrechten verwaltet werden.



Im Formular *Registered Customers* werden die Kundendaten verwaltet. Für jeden Kunden werden die Registrierungsnummer, der Aktivierungsschlüssel und die gewährten Rechte gespeichert. So ist es erforderlichenfalls einfach möglich einen neuen Aktivierungsschlüssel zu erstellen und zu versenden.

Die Schaltfläche *Create activation key* öffnet den Dialog zur Generierung eines Aktivierungsschlüssels. Nach Erstellen eines Aktivierungsschlüssels wir die Kundenliste automatisch aktualisiert.

Wenn der Anwender seinen Installationsschlüssel sendet, muss ein Aktivierungsschlüssel erstellt werden. Dieser Aktivierungsschlüssel teilt der Anwendung mit, ob der Anwender eine bestimmte Aktion ausführen darf. Für jede Aktion muss das entsprechende Recht ausgewählt werden.



Mit der Schaltfläche „Read Installation Key“ öffnet sich ein Dialog, in den der Installationsschlüssel des Anwenders eingegeben wird. Der Installationsschlüssel kann über die Zwischenablage eingefügt werden oder aus einer Datei gelesen werden.

Nachdem jedes für den Anwender erlaubte Recht markiert ist, wird mit einem Klick auf OK der Aktivierungsschlüssel generiert. Der erstellte Aktivierungsschlüssel wird in der Datei *<Projektname>.xak* im Projektordner gespeichert. Der Aktivierungsschlüssel oder die Datei muss an den Anwender zur Aktivierung der Anwendung gesendet werden.

Wenn dem Anwender entsprechend dem obigen Beispiel alle Rechte zur Datenbearbeitung gegeben wurden, er aber nicht das Recht hat Berichte zu drucken, sehen die Eigenschaften zur Laufzeit so aus:

```
goProgram.SecurityRights.RunDataForms = 1
goProgram.SecurityRights.RunReports = 0
goProgram.SecurityRights.EditData = 1
goProgram.SecurityRights.ViewData = 1
```

Wenn der Anwender eine Anwendung startet, die eine Aktivierung erfordert (und wenn die Anwendung noch nicht aktiviert wurde), wird automatisch der Installationsschlüssel erzeugt. Abhängig vom Wert der Eigenschaft *nRegWay* wird der Installationsschlüssel entweder angezeigt oder in einer Datei gespeichert, die per E-Mail versendet werden kann. Nachdem der Anwender den Aktivierungsschlüssel erhalten hat, kann er ihn im Aktivierungsfenster eingeben oder die Datei mit dem Aktivierungsschlüssel im Projektordner speichern. Damit ist die Anwendung auf diesem Computer aktiviert.

Wenn der Anwender später den Menüpunkt *Hilfe, Produkt aktivieren* auswählt, wird der Installationsschlüssel angezeigt, unabhängig von der Einstellung der Eigenschaft *nRegWay*.

11.5. Eigenschaften der Klasse CVFXActivation

cFirstInstall – Diese Eigenschaft enthält den Namen einer Datei. Anhand des Vorhandenseins dieser Datei entscheidet diese Klasse, ob die Anwendung erstmalig gestartet wird. Wenn dieser Eigenschaft eine leere Zeichenkette zugewiesen wird, kann nicht überprüft werden, ob die Anwendung erstmalig gestartet wird. Das Datum des Starts wird dann ohne weitere Überprüfung in der Ini-Datei eingetragen.

cINIFileName – Der Name der Ini-Datei, in der die Aktivierungsinformationen und das Datum des ersten Anwendungsstarts gespeichert sind. Der Standardwert ist *VFX.ini*.

cParamFile – Der Name der Datei, in der der Installationsschlüssel gespeichert wird. Abhängig vom Wert der Eigenschaft *nRegWay* kann diese Datei per E-Mail versendet oder auf einem anderen Weg verarbeitet werden.

cRegMail – In dieser Eigenschaft wird die E-Mail-Adresse des Entwicklers gespeichert, an die die Datei mit dem Installationsschlüssel gesendet wird, wenn die Eigenschaft *nRegWay* den Wert 2 hat.

cRegFileName – Hier kann der Name einer Datei angegeben werden, die bei der Installation erstellt wird. Das Erstellungsdatum dieser Datei wird verwendet um das Installationsdatum zu ermitteln. Wenn dieser Eigenschaft kein Wert zugewiesen wird, wird das Systemdatum des ersten Starts der Anwendung verwendet.

nRegWay – In dieser Eigenschaft kann eingestellt werden, wie der Entwickler den Installationsschlüssel bekommen soll.

0 – Der Installationsschlüssel wird in einem Dialog angezeigt und der Anwender kann den Installationsschlüssel kopieren und in beliebige Anwendungen einfügen.

1 – Der Installationsschlüssel wird in einer Datei gespeichert. Der Anwender kann diese Datei später an den Entwickler übermitteln. Der Name der Datei wird in der Eigenschaft *cParamFile* hinterlegt.

2 – Der Installationsschlüssel wird in einer Datei gespeichert und an den Entwickler als E-Mail-Anhang gesendet. Der Name der Datei wird in der Eigenschaft *cParamFile* hinterlegt. Die E-Mail-Adresse des Entwicklers, an die der Installationsschlüssel gesendet wird, wird in der Eigenschaft *cRegEMail* eingetragen.

11 – Nach Anzeige des Registrierungsdialogs wird der Installationsschlüssel in einer Datei gespeichert. Diese Datei kann später an den Entwickler gesendet werden. Der Dateiname wird in der Eigenschaft *cParamFile* hinterlegt.

12 – Nach Anzeige des Registrierungsdialogs wird der Installationsschlüssel in einer Datei gespeichert und sofort als E-Mail-Anhang an den Entwickler geschickt. Der Dateiname muss in der Eigenschaft *cParamFile* hinterlegt werden. Die E-Mail-Adresse des Entwicklers muss in der Eigenschaft *cRegEMail* eingetragen werden.

12. Erstellen mehrsprachiger Anwendungen

VFX ist gut vorbereitet, um mehrsprachige Anwendungen zu erstellen. Sie können zwischen Lokalisierung während der Entwicklung und Lokalisierung zur Laufzeit wählen.

12.1. Lokalisierung zur Entwicklungszeit

Bei der Erstellung eines neuen VFX-Projekts kann zwischen verschiedenen Sprachen gewählt werden. Entsprechend der gewählten Sprache werden Include-Dateien für die gewählte Sprache im neuen Projekt generiert.

Will man zu einem späteren Zeitpunkt seine Anwendung in eine andere Sprache übersetzen, startet man für jedes Formular den VFX – LangSetup Builder. Dieser Builder erstellt für jede Caption eines Formulars eine Zuweisung. Der Caption wird zur Laufzeit der Wert einer Konstanten zugewiesen.

Die Konstanten können mit dem VFX – Message Editor bearbeitet werden. Zur Erstellung der Anwendung kopiert man dann einfach die Include-Dateien der gewünschten Sprache in das Projekt und lässt die Anwendung neue erstellen.

Die Bedienungselemente tauchen in den folgenden Bereichen auf:

- Bedienung der bestehenden Funktionalität in den Visual Extend-Klassenbibliotheken und allen Dialogen.
- Bedienung Ihrer eigenen Anwendung.

Sie brauchen sich nicht um den ersten Punkt zu kümmern.

Die Bedienungselemente der bestehenden Funktionalität in den Visual Extend-Klassenbibliotheken und allen Dialogen existieren in mehreren Sprachen. Sie brauchen kein Wort zu übersetzen, wenn Ihre Anwendung in einer der zur Verfügung stehenden Sprachen erstellt werden soll. Wenn Sie die Visual Extend-Klassenbibliotheken in einer anderen Sprache benötigen, können Sie die Tabelle *Vfxmsg.dbf* selbst erweitern.

Wir wären Ihnen sehr dankbar, wenn Sie uns Ihre Übersetzung der VFX-Meldungen in der Tabelle *Vfxmsg.dbf/cdx/fpt* in eine noch nicht vorhandene Sprache zusenden würden. Wir könnten diese dann anderen Entwicklern zur Verfügung stellen. Vielen Dank!

Prüfliste für die Erstellung mehrsprachiger Anwendungen mit VFX:

√ Benutzen Sie die Include-Dateien *USERTXT.H* bzw. *USERMSG.H*, die vom **VFX – Message Editor** erstellt werden, um alle sprachabhängigen Bedienungselemente für Ihre Anwendung zu verwalten. ***Der Speicher für Bezeichnungen, Meldungen, Überschriften, Tooltip-Texte und Statuszeilenmeldungen ist die Tabelle VFXMSG.DBF. In dieser Tabelle finden Sie auch alle von VFX benutzten Texte, die bereits in die zur Verfügung stehenden Sprachen übersetzt sind.***

√ Benutzen Sie in Ihrer Anwendung Konstanten anstelle von direkten Texten, z. B. **WAIT WINDOW Loc_Text1** anstelle von **WAIT WINDOW "MyText…".**

√ Benutzen Sie die Include-Datei *USERDEF.TXT* für alle anwendungsspezifischen Konstanten, die sprachunabhängig sind. Dadurch wird Ihre Lokalisierungsarbeit erleichtert.

√ Benutzen Sie den **VFX – LangSetup Builder,** um den Code für die VFX-Formularmethode mit dem Namen *LangSetup()* zu erstellen. Die Methode enthält den Lokalisierungscode. Den Bezeichnungen, Tooltip-Texten usw. werden die Werte aus den Konstanten zugewiesen. (Der VFX – LangSetup Builder erzeugt automatisch den Code für die *LangSetup()*-Methode und aktualisiert die Tabelle *VFXMSG.DBF* mit den Meldungen und Bezeichnungen.)

√ Übersetzen Sie Ihren Text mit dem ***VFX – Message Editor*** in die verschiedenen Sprachen. Der VFX-Message-Editor erzeugt Include-Dateien für die verschiedenen Sprachen im Ordner *\INCLUDE\LanguageDir*. *LanguageDir* steht für den Namen der Sprache, in die Sie übersetzen. (Wie oben bereits erwähnt, wurden die VFX-spezifischen Sprachkonstanten bereits in einige Sprachen übersetzt. Sie brauchen hierfür kein einziges Wort zu übersetzen.)

√ Um Ihre Anwendung für eine Sprache zu erstellen, definieren Sie die Konstante *ID_LANGUAGE* in der *VFXDEF.H*-Include-Datei und kopieren Sie die Include-Datei aus dem Ordner *\INCLUDE\LanguageDir* in den aktuellen *\INCLUDE*-Ordner Ihres Projektes.

√ Wählen Sie die Option *Alle Dateien neu kompilieren* und testen Sie Ihre Anwendung. Sie erhalten für jede Sprache eine eigene EXE-Datei.

12.2. Lokalisierung zur Laufzeit

Mit VFX 11.0 können nicht nur Anwendungen für verschiedene Sprachen lokalisiert erstellt werden, es ist jetzt auch möglich die Sprache einer Anwendung zur Laufzeit umzustellen.

Die Möglichkeit zur Umstellung der Sprache zur Laufzeit wird über die Eigenschaft *goProgram.lRuntimeLocalization* des Anwendungsobjekts gesteuert. Wenn dieser Eigenschaft der Wert .T. zugewiesen wird, kann die Sprache der Anwendung im Anmeldedialog ausgewählt werden. Zusätzlich kann, während die Anwendung läuft, die Sprache über eine Combobox in der Standard-Symbolleiste umgeschaltet werden.

Die Eigenschaft *goProgram.lRuntimeLocalization* kann mit dem VFX Application Builder eingestellt werden.



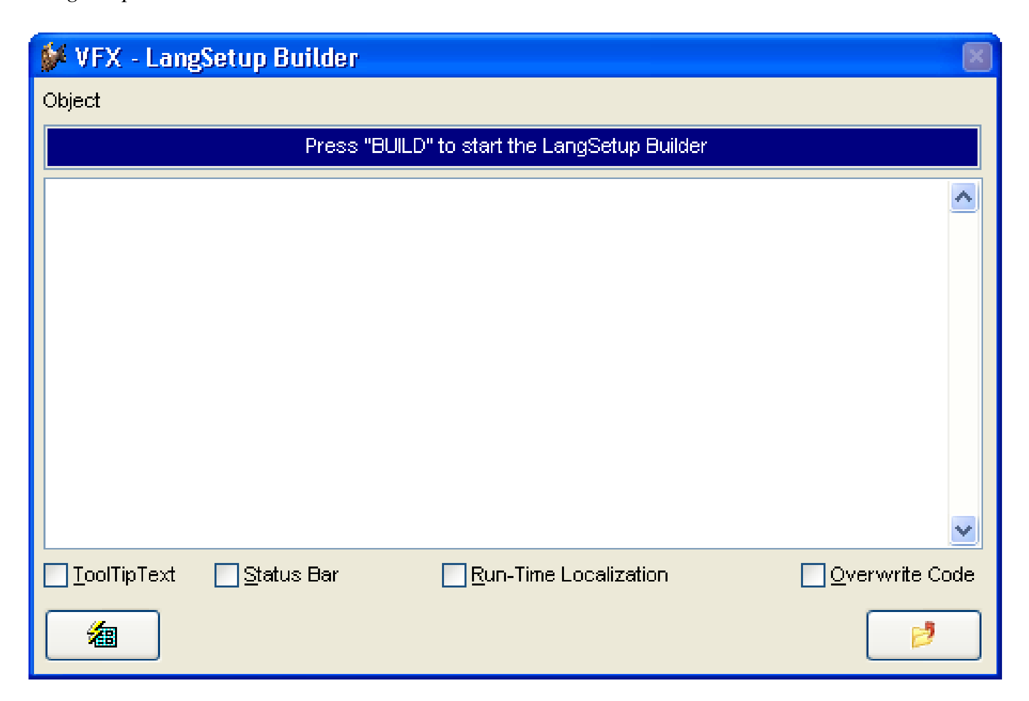
Wenn die Lokalisierung zur Laufzeit aktiviert wird ist, wird ein global sichtbares Objekt mit dem Namen *goLocalize* beim Anwendungsstart instanziiert. Dieses Objekt hat Eigenschaften entsprechend den Texten in der Tabelle *Vfxmsg.dbf*. Für jeden Datensatz in der Tabelle *Vfxmsg.dbf* wird dem Objekt *goLocalize* zur Laufzeit eine Eigenschaft hinzugefügt. Der Name der Eigenschaft entspricht der *Message_ID* mit dem Präfix *c*. Wenn sich beispielsweise in der Tabelle *Vfxmsg.dbf* ein Datensatz mit der *Message_ID CAP_APPLICATION_TITLE* befindet, heißt die entsprechende Eigenschaft des Lokalisierungsobjekts *goLocalize.CCAP_APPLICATION_TITLE*. Auf das Lokalisierungsobjekt und seine Eigenschaften kann jederzeit zugegriffen werden.

Die von jedem Benutzer zuletzt verwendete Sprache wird in der Ressourcentabelle *Vfxres.dbf* gespeichert. Wenn sich ein Benutzer erneut anmeldet, erscheint die Anwendung in der zuletzt benutzten Sprache.

12.3. VFX – LangSetup Builder

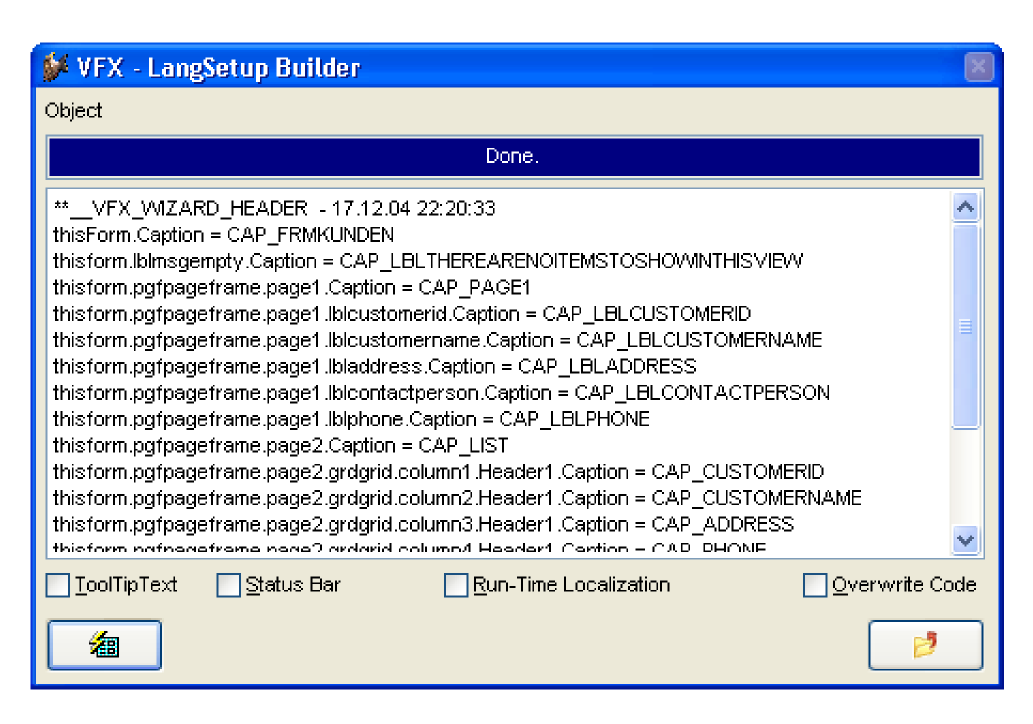
Der VFX – LangSetup Builder automatisiert die Erstellung des in der *LangSetup()*-Methode benötigten Codes. Sie brauchen diesen Code, wenn Sie Ihre Anwendung in mehr als einer Sprache erstellen wollen. Das Ziel dieses Builders ist es, aus dem Formular für alle Bezeichnungen, Tooltip-Texte und Statuszeilenmeldungen Datensätze anzulegen und diese in der Tabelle *Vfxmsg.dbf* zu speichern. Nach diesem Vorgang können Sie den VFX – Message Editor benutzen, um die Texte in verschiedene Sprachen zu übersetzen.

Um den VFX – LangSetup Builder aufzurufen, öffnen Sie zunächst das Formular dessen Bezeichnungen, Tooltip-Texte und Statuszeilenmeldungen Sie analysieren lassen möchten. Wählen Sie den Menüpunkt *Form, LangSetup Builder* aus dem VFX-Menü:



Markieren Sie die Kontrollkästchen entsprechend den gewünschten Optionen. Klicken Sie auf die Schaltfläche *Build* um den Code für die *LangSetup()*-Methode generieren zu lassen.

Nach der Generierung sehen Sie den Code, der für die *LangSetup()*-Methode erzeugt wurde. Wenn Sie das Kontrollkästchen *Overwrite Code* markieren, wird der erzeugte Code in die *LangSetup()*-Methode des aktuell in der Entwicklungsansicht geöffneten Formulars geschrieben. Der Bezeichnungscode wird in der VFX-Meldungstabelle *Vfxmsg.dbf* gespeichert. Hier können Sie die Texte bearbeiten und in andere Sprachen übersetzen.



In der Include-Datei *VFX.h* gibt die Konstante *_LANG_SETUP* an, ob die *LangSetup()*-Methode ausgeführt wird. In der *LangSetup()*-Methode wird überprüft, ob diese Konstante existiert und falls ja, wird der Code der Methode ausgeführt. Dieses Verfahren dient der Geschwindigkeitsoptimierung für die Formulare.

```
#DEFINE _LANG_SETUP            .T.
```

In der Include-Datei *Vfxdef.h* ist die *ID_Language*-Konstante definiert, die die aktuelle Sprache Ihrer Anwendung angibt.

```
…
#define ID_LANGUAGE "ENG"
…
```

Wenn Sie Ihre Anwendung mit dem VFX-Anwendungs-Assistenten anlegen, wird die Anwendung in der Sprache angelegt, die im VFX-Anwendungs-Assistenten angegeben ist. Wenn Ihre Anwendung in eine andere Sprache übersetzt werden soll, ändern Sie die Konstante *ID_Language*.

13. VFX.fll

Die Datei *VFX.fll* enthält zahlreiche Funktionen, die für die Produktaktivierung, die Datensicherung sowie für den Zugriff auf SQL Server und auf das Internet benötigt werden. Die *VFX.fll* muss zusammen mit den Anwendungen an die Kunden ausgeliefert werden. Die Funktionen der *VFX.fll* werden im Einzelnen beschrieben.

13.1. Produktaktivierung

GetAppRights(lcRightsBin, This.Hex2Bin(This.cActPattern)) – Liefert Informationen über ein Recht aus der Produktaktivierung. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CVFXActivate* in der Methode *checkactstate*.

Rückgabewert:
0 – Der Vorgang wurde erfolgreich ausgeführt.
-1 – Die Länge des Aktivierungsschlüssels ist ungültig.
-2 – Der Aktivierungsschlüssel ist inkonsistent.
-3 – Fehler bei der Verschlüsselung.

GetFileCreationDateTime(cFileName) – Liefert Datum und die Uhrzeit zu der eine Datei erstellt wurde. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CVFXActivate* im Ereignis *Init()*.

cFileName – Name der zu überprüfenden Datei.

Rückgabewert: Ein Zeit/Datum-Wert als Zeichenkette.

GetSysInfo(This.Hex2bin(This.cActPattern)) – Diese Funktion liefert den Installationsschlüssel. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CVFXActivate* in der Methode *checkactstate*.

13.2. Datensicherung oder Archivierung

CreateZipArchive(tcPath, tcFileMask, tcArchiveFullPathName, tcFeedBackFunction, tnCompressionLevel, tlRecurseSubfolders, tcPassword) – Erstellen einer Zip-Archivdatei. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CArchive* in der Methode *createarchive*.

tcPath – Pfad des zu archivierenden Ordners.

tcFileMask – Namen der zu archivierenden Dateien. Es kann mit Platzhalterzeichen gearbeitet werden. Mehrere Dateinamen können durch Semikolon getrennt aufgeführt werden.

tcArchiveFullPathName – Pfad- und Dateiname des zu erstellenden Zip-Archivs.

tcFeedBackFunction – Name einer Funktion oder Methode, die von *CreateZipArchive* aufgerufen wird und Informationen über den Fortschritt zu liefern.

tcFeedBackFunction(cCurrentOperatedFile, nState, nAllFilesSize, nZIPedFilesSize, nArchiveCurrentSize) – Diese Funktion oder Methode wird von *CreateZipArchive* immer dann aufgerufen wenn die zu erstellende Zip-Datei bereits existiert,
bevor eine Datei dem Archiv hinzugefügt wird,
nachdem eine Datei dem Archiv hinzugefügt wurde,
nachdem ein Archiv erfolgreich erstellt wurde,
wenn ein Archiv nicht erstellt werden konnte,
eine Datei nicht dem Archiv hinzugefügt werden konnte.

cCurrentOperatedFile – Name der Datei, die zurzeit bearbeitet wird.

nState – Status.

1 – Die Datei *cArchiveFullPathName* existiert bereits.
2 – Beginn des Hinzufügens der Datei *cCurrentOperatedFile* zum Archiv.
3 – Ende des Hinzufügens der Datei *cCurrentOperatedFile* zum Archiv.
4 – Die Datei *cCurrentOperatedFile* konnte dem Archiv nicht hinzugefügt werden.
5 – Die Erstellung des Archivs wurde vollständig abgeschlossen.
6– Die Erstellung des Archivs konnte nicht abgeschlossen werden.
7 – Es wurde kein gültiger Pfad- oder Dateiname angegeben bzw. es sind keine Dateien zu archivieren.

nAllFilesSize – Gesamtgröße aller Dateien, die dem Archiv hinzugefügt werden sollen.

nZIPedFilesSize – Größe der Dateien, die dem Archiv bereits hinzugefügt wurden.

nArchiveCurrentSize – Momentane Größe der erstellten Archivdatei.

Rückgabewert:
0 – Der Vorgang wurde abgebrochen.
1 – Die Dateien wurden dem Archiv hinzugefügt.
2 – Der Vorgang wird fortgesetzt.

tnCompressionLevel – Der ZIP-Algorithmus erlaubt verschiedene Komprimierungsstufen. Als Werte sind -1 bis 9 erlaubt. Die Werte bedeuten:

-1 – Standardkomprimierung
0 – keine Komprimierung
1 – höchste Geschwindigkeit
6 – Standardkomprimierung
9 – beste Komprimierung

Die hier nicht aufgeführten Werte erlauben eine Feinstellung und so einen Kompromiss zwischen Geschwindigkeit und Komprimierung. Die Standardkomprimierung kann wahlweise mit dem Wert -1 oder mit dem Wert 6 erreicht werden.

tlRecurseSubfolders – Wenn der Wert dieses Parameters *True* ist, werden Unterordner rekursiv mit eingeschlossen. Die als *tcFileMask* gewählten Dateien werden auch in den Unterordnern berücksichtigt. Wenn der Wert dieses Parameters *False* ist, werden Unterordner nicht mit eingeschlossen.

tcPassword – Hier muss ein Kennwort eingegeben werden, wenn das Archiv geschützt werden soll. Wenn kein Kennwortschutz benötigt wird, muss hier eine leere Zeichenkette übergeben werden. Für das Kennwort sind alle Zeichen, außer CHR(0) zulässig.

ExtractZipArchive(tcExtractFilesFolder, tcFileMask, tcArchiveFullPathName, tcFeedBackFunction, tcPassword) Entpacken von Dateien aus einer Zip-Archivdatei. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CArchive* in der Methode *extractfromarchive*.

tcExtractFilesFolder – Ordner, in den die entpackten Dateien gespeichert werden.

tcFileMask – Namen der zu entpackenden Dateien. Mehrere Dateinamen können durch Semikolon getrennt angegeben werden. Es kann mit Platzhalterzeichen gearbeitet werden.

tcArchiveFullPathName - Name und Pfadname der Archivdatei.

tcFeedBackFunction – Name einer Funktion oder Methode, die aufgerufen wird um Informationen über den Fortschritt zu liefern.

cFeedBackFunction(cCurrentOperatedFile, nState, nArchiveFilesSize, nUnZIPedFilesSize) – Diese Funktion oder Methode wird von *cFeedBackFunction* immer dann aufgerufen wenn eine zu entpackende Datei bereits existiert,
das Entpacken einer Datei beginnt,

das Entpacken einer Datei endet,
eine Datei nicht aus dem Archiv entpackt werden kann,
das Entpacken aller Dateien erfolgreich abgeschlossen wurde,
das Entpacken aller Dateien nicht abgeschlossen werden konnte.

cCurrentOperatedFile – Name der zurzeit entpackten Datei.

nState Status
1 – Die zurzeit bearbeitete Datei existiert bereits.
2 – Beginn des Entpackens der Datei *cCurrentOperatedFile*.
3 – Ende des Entpackens der Datei *cCurrentOperatedFile*.
4 – Die Datei *cCurrentOperatedFile* konnte nicht entpackt werden.
5 – Der Vorgang wurde erfolgreich abgeschlossen.
6 – Der Vorgang konnte nicht abgeschlossen werden.

Rückgabewert:
0 – Abbruch des Entpackens.
1 – Fortsetzen des Vorgangs.
2 – Überschreiben der bestehenden Datei mit der Archivdatei.

tcPassword – Kennwort zum Entpacken des Archivs, falls benötigt. Wenn kein Kennwort zum Entpacken erforderlich ist, muss eine leere Zeichenkette übergeben werden.

13.3. SQL Server

GetSQLServers(@cServersString, @cEerrorString) – Ermitteln aller verfügbaren SQL Server. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Funktion *TryConnecting* in *Vfxfunc.prg*.

cServersString – Zeichenkette, die eine durch Komma getrennte Liste mit den Namen aller verfügbaren SQL Server enthält.

cErrorString – Eventuell aufgetretene Fehler werden hier zurückgegeben.

Rückgabewert: Anzahl der ermittelten SQL Server.

GetSQLDataBases(cServer, @cDBString, cUser, cPass, @cErrors) – Ermitteln aller Datenbanken eines SQL Servers.

cServer – Name des SQL Servers von dem die Datenbanken ermittelt werden sollen.

cDBString – Eine Zeichenkette mit den durch Komma getrennten Namen aller verfügbaren Datenbanken.

cUser – Benutzername für die Anmeldung beim SQL Server.

cPass – Kennwort für die Anmeldung beim SQL Server.

cErrors – Eventuelle Fehlermeldung des SQL Servers.

Rückgabewert: *0* – Der Vorgang wurde erfolgreich abgeschlossen.

13.4. Internet, E-Mail und Hilfsfunktionen

URLDownload2File(cUrl, cFileName, cFeedBackFunction, cCancelDownload) – Download einer Datei aus dem Internet. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CDownload* in der Methode *download*.

cUrl – URL der Datei, die heruntergeladen werden soll.

cFileName – Datei- oder Pfadname. Hier wird die heruntergeladene Datei gespeichert.

cFeedBackFunction – Name einer Funktion oder Methode, die von *URLDownload2File* aufgerufen wird, um Informationen über den Fortschritt zu liefern. Die Funktion oder Methode muss zwei Parameter akzeptieren.
cFeedBackFunction(nCurrentAmount, nFileSize)
nCurrentAmount – Anzahl der bereits heruntergeladenen Bytes.
nFileSize – Größe der herunterzuladenden Datei.

cCancelDownload – Name einer Variablen oder Eigenschaft, die den Fortgang des Downloads steuert. Die Variable oder Eigenschaft wird automatisch ständig überprüft.
cCancelDownload = .F. – Der Download wird fortgesetzt.
cCancelDownload = .T. – Der Download wird abgebrochen.

Rückgabewert: *0* – Der Download wurde erfolgreich abgeschlossen.

Get_PS_Printers(nLocation, @cPrinterNames, @nPrinterNamesLength) – Liefert die Namen aller installierten Postscript-Druckertreiber. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CCreatePDF* in der Methode *checkpsprinter*.

nLocation – Standort des Druckers.
1 – Es wird nach lokalen Druckern gesucht.
2 – Es wird nach Netzwerkdruckern gesucht.
3 – Es wird lokalen Druckern und Netzwerkdruckern gesucht.

cPrinterNames – Enthält die Namen aller installierten Postscript-Druckertreiber in einer Komma-separierten Liste.

nPrinterNamesLength – Länge der zurückgegebenen Zeichenkette.

Rückgabewert: *0* – Der Vorgang wurde erfolgreich ausgeführt.

Add_Printer(cPrinterName ,cPrinterPort) – Vollautomatische Installation eines Druckertreibers. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CCreatePDF* in der Methode *checkpsprinter*.

cPrinterName – Name des zu installierenden Druckertreibers.

cPrinterPort – Anschluss des zu installierenden Druckertreibers.

Rückgabewert: *0* – Die Installation wurde erfolgreich abgeschlossen.

Encrypt(cStringForEncrypting, cPassword) – Verschlüsselung einer Zeichenkette mit einem Kennwort. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CDunConnectiom.cmdOk* im Ereignis *Click()*.

cStringForEncripting – Zu verschlüsselnde Zeichenkette.

cPassword – Das zur Verschlüsselung dienende Kennwort.

Rückgabewert: Verschlüsselte Zeichenkette.

Decrypt(cStringForDecripting ,cPassword) – Entschlüsselung einer Zeichenkette mit einem Kennwort. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CDunConnection* im Ereignis *Init()*.

cStringForDecripting – Zu entschlüsselnde Zeichenkette.

cPassword – Das zur Entschlüsselung dienende Kennwort.

Rückgabewert: Entschlüsselte Zeichenkette.

GetAxControlSize(nhWnd, @nWidth, @nHeight) – Rückgabe der Größe eines ActiveX-Steuerelements. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CCalendar* in der Methode *Resize()*.

nhWnd – Handle des Fensters des ActiveX-Steuerelements.

nWidth – Breite des ActiveX-Steuerelements.

nHeight – Höhe des ActiveX-Steuerelements.

Rückgabewerte:
.T. – Die Größe des ActiveX-Steuerelements konnte erfolgreich ermittelt werden.
.F. – Die Größe des ActiveX-Steuerelements konnte nicht ermittelt werden.

SetModemConnection(cConnectionName, cPhoneNumber, cUserName, cPassword) – Einrichten einer DFÜ-Netzwerkverbindung. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CDownload* in der Methode *establishdunconnection*. Für die erfolgreiche Ausführung dieser Funktion muss ein Modemtreiber installiert sein!

cConnectionName – Name der zu erstellenden DFÜ-Netzwerkverbindung.

cPhoneNumber – Zu wählende Rufnummer.

cUserName – Benutzername der Verbindung.

cPassword – Kennwort der Verbindung.

Rückgabewert:
.T. – Die DFÜ-Netzwerkverbindung wurde erfolgreich angelegt.
.F. – Die DFÜ-Netzwerkverbindung konnte nicht angelegt werden.

CheckInetConn(cCheckURL, cDUNConnName, nHWnd) – Diese Funktion überprüft, ob eine Verbindung mit dem Internet besteht. Hierzu wird eine URL im Internet aufgerufen. Ein Beispiel für die Anwendung dieser Funktion befindet sich in der Klasse *CDownload* in der Methode *checkinternetconnection*.

cCheckURL – Diese URL wird überprüft um festzustellen, ob eine Verbindung mit dem Internet besteht.

cDUNConnName – Über diese DFÜ-Netzwerkverbindung wird bei Bedarf eine Verbindung hergestellt.

nHWnd – Handle des aufrufenden Fensters.

Rückgabewerte:
0 – Es besteht eine Verbindung mit dem Internet.
-1 – Die Verbindungsherstellung wurde durch den Benutzer abgebrochen.
-2 – Es besteht keine Verbindung mit dem Internet.
-3 – Es ist ein Fehler aufgetreten.
24 – Die DFÜ-Netzwerkverbindung mit dem Namen *cDUNConnName* existiert nicht.

14. VFX – AFP Wizard

Dieser Wizard erzeugt aus bestehenden VFX 11.0 Formularen lauffähige aktive AFP Webseiten.

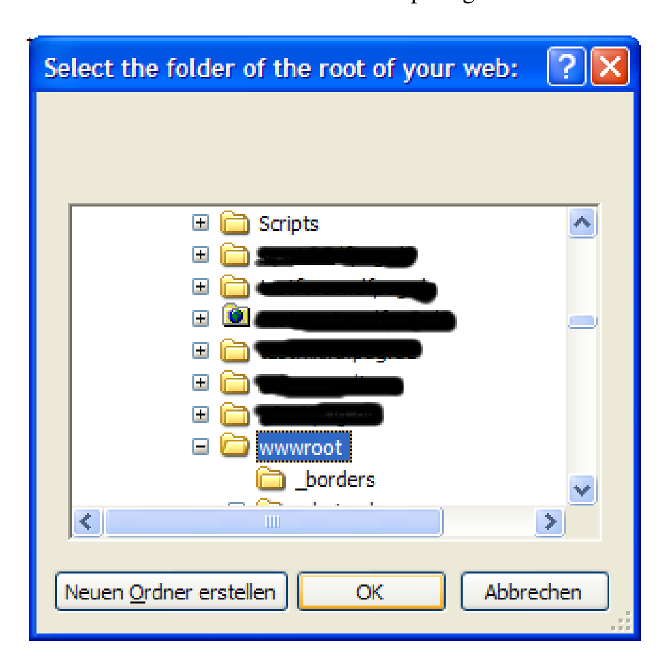
Eine aktuelle Version der AFP (Active Foxpro Pages) finden Sie unter http://www.afpages.de.

Der Wizard unterstützt zurzeit nur Formulare, welche mit DBF-Tabellen arbeiten. Cursoradapter werden in einer zukünftigen Version unterstützt.

Der Wizard funktioniert mit Formularen, die auf einer der VFX-Formularklassen cdataformpage oder ctableform basieren. Weitere VFX-Formularklassen werden in einer in späteren Version unterstützt.

Beim ersten Start des Wizards wird die verwendete Metadatentabelle vfxafpmeta.dbf unter C:\Dokumente und Einstellungen\All Users\Anwendungsdaten\dFPUG\Visual Extend\11.0 abgelegt.

Der Pfad für die Ausgabe der erzeugten AFP-Seiten wird aus der Registry HKLM\SOFTWARE\Microsoft\InetStp ausgelesen und zur Auswahl angeboten.

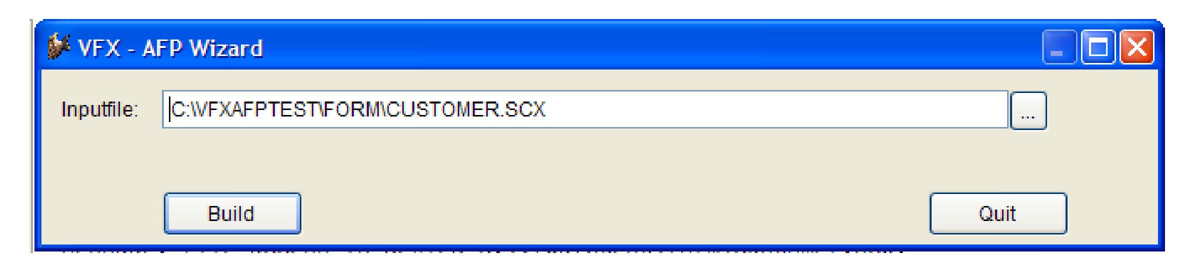


Geben Sie hier den Pfad Ihres lokalen Webs an, in dem die Dateien abgelegt werden sollen.

Bei jedem Lauf des Wizards wird automatisch überprüft ob die noch zusätzlichen notwendigen Dateien vorhanden sind. Bei Bedarf werden diese automatisch angelegt.

Die Verzeichnisse lauten vfxafpstyle für die stylesheets, vfxafpimage für die Bilder und vfxafpjs für das Javascript, welches in den Grids zurzeit verwendet wird.

Nun erscheint der Wizard.



Wählen Sie Ihr VFX Formular aus und klicken Sie auf Build.

Anmerkung: Das zuletzt verwendete Formular wird automatisch angezeigt.

Es wird der Anmeldeschirm erscheinen, genau so als ob sie die Applikation gestartet hätten.

Die AFP-Seiten werden erzeugt und können dann unter

http://localhost/meinverzeichnis/frm_formularname.afp

gestartet werden.

Im Fehlerfall:
Der Fehler, der auftaucht, wenn man eine Maske (gravierend) ändert und dann sofort den Builder startet, ist „`Error loading Form`"
Nachdem man den Anmeldeschirm verlassen hat. Dies liegt an der Resourcedatei, welche zuerst mit der Benutzerverwaltung im laufenden Programm gelöscht werden muss.

Starten Sie Ihre Anwendung, melden Sie sich an und gehen sie unter Benutzerverwaltung auf die Seite *Bearbeiten*.

Dort können Sie die Schaltfläche „Einstellungen Löschen" anklicken.

14.1. Beschreibung der vfxafpmeta.dbf

Die Tabelle hat folgende Felder:

ckey	beinhaltet den Klassennamen
cdesc	eine kurze Beschreibung
cmemo	der Inhalt bzw. der HTML Code
lparam	.T. bedeutet dies ist ein Parameter zur Ablaufsteuerung
lCode	.T. bedeutet, dass der Inhalt von cmemo per execscript ausgeführt wird
nvers	die aktuelle Versionsnummer

Es gibt 5 Parameter:

Outputpath	Der Pfad, welcher beim ersten Start des Wizards eingegeben werden muss.
Prefix	Der Prefix, welcher vor jedem Formularnamen vorangestellt wird. Default="frm_"
Postfix	Der Postfix, welcher dem Formularnamen angehängt wird.
Extension	die Extension der erzeugten Dateien. Default=".AFP"
Postfixexec	der Postfix für die EXEC-Dateien, welche den Code enthalten um die Eingaben abzuarbeiten.

Jede verwendete Klasse im Formular wird mit zwei Datensätzen abgebildet.

Am einfachsten zu Erklären ist dies mit der Pageframe, welche aus Pageframe und Page besteht.

Innerhalb einer Pageframe können beliebig viele Pages liegen. Also muss die Pageframe am Ende auch geschlossen werden.

Der Anfangscode liegt also im Datensatz pageframe dann kommt der Datensatz Page nun alle darin enthaltenen Elemente, wie Textboxen oder Labels und nun müssen mit Page_end und Pageframe_end die den Endecode enthalten.

Im Fall der pageframe ist dies

```
<div  id="<<cname>>" class="pageframe" style=" position: relative;
width: <<nwidth>>px;height: <<nheight>>px; z-index:<<nlevel>>; left:
<<nleft>>px; top: <<ntop>>px" >
```

Und in pageframe_end steht dann nur noch

```
</div>
```

So wird mit jedem Objekt, jeder Klasse verfahren.

Ist eine Klasse nicht gefüllt, so wird automatisch die Basisklasse gesucht und herangezogen. Dadurch ist eine kleine Objektorientiertheit angedacht.

15. Weitere Entwicklungstechniken

15.1. Hinzufügen eines Formulars zum Öffnen-Dialog

VFX bietet einen Öffnen-Dialog zum Öffnen von Formularen. Selbstverständlich können Sie diesen Dialog an Ihre Bedürfnisse anpassen oder einen eigenen Dialog erstellen.

Zusätzlich zu dem in bisherigen VFX-Versionen vorhandenem Öffnen-Dialog (*Vfxfopen.scx*) steht in VFX 11.0 ein neuer Öffnen-Dialog im Windows-XP-Stil (*Vfxxpopen.scx*) zur Verfügung. Dieser neue Öffnen-Dialog ist standardmäßig aktiviert. Mit der Eigenschaft *goprogram.lxpopenstyle* kann auf Wunsch auf den alten Öffnen-Dialog umgeschaltet werden.



lxpopenstyle

.T. – der neue Öffnen-Dialog im Windows-XP-Stil wird verwendet.

.F. – der alte Öffnen-Dialog (*Vfxfopen.scx*) wird verwendet.

Die Gruppenüberschriften im neuen Öffnen-Dialog werden aus dem neuen Tabellenfeld *Vfxopen.groupcap* gelesen. Der Zustand der einzelnen Gruppen (aufgeklappt oder zugeklappt) wird je Benutzer gespeichert.

Der Datei/Öffnen-Dialog benutzt die Tabelle *Vfxfopen.dbf*. Die VFX-Formular-Builder fügen automatisch für jedes Formular einen Datensatz zu der Tabelle *Vfxfopen.dbf* hinzu. Hier ist die Struktur der Tabelle *Vfxfopen.dbf*:

VFXFOpen-Feld	***Beschreibung***	***Beispiel***
ObjectID	Dieses Feld wird verwendet, wenn der Öffnen-Dialog *Vfxfopen.scx* verwendet wird. Hierzu muss die Eigenschaft *goProgram.lxpopenstyle=.F.* gesetzt sein. Der VFX-Öffnen-Dialog hat normalerweise zwei Seiten. (**Tipp:** Sie können die *Pagecount*-Eigenschaft des Seitenrahmens im Formular *Vfxopen.scx* auf jeden beliebigen Wert setzen, um die Anzahl der Seiten zu verändern.) Wenn Sie wollen, dass Ihr Formular auf Seite 1 des Seitenrahmens erscheint, geben Sie PAGE1 ein. Für die weiteren Seiten PAGE2, PAGE3 usw.	PAGE1
ObjectNo	Geben eine Zahl für die Sortierfolge der Liste ein. 1 wird das erste Element, es folgt 2 usw. Die Sortierung wird auf jeder Seite benutzt.	1
GroupCap	Dieses Feld wird verwendet, wenn der Öffnen-Dialog *Vfxxpopen.scx* verwendet wird. Hierzu muss die Eigenschaft *goProgram.lxpopenstyle=.T.* gesetzt sein. Dieses Feld enthält eine Gruppenüberschrift. Die Gruppierung erfolgt entsprechend der Einträge im Feld *ObjectID*. Die *GroupCap* muss nur für den ersten Eintrag einer Gruppe eingetragen werden.	Kontakte
Title	Geben Sie die Überschrift ein, die im Listenfenster erscheint.	Kunden
Descr	Geben Sie einen Beschreibungstext ein, der angezeigt wird, wenn der Benutzer diesen Eintrag ausgewählt hat.	Liste aller Adressen
Form	Geben Sie den Namen des aufzurufenden Formulars ein.	ADRE
Parameter	Wenn Sie an das Formular Parameter übergeben wollen, können Sie diese hier eingeben.	
Viewlevel	Die Benutzerstufe, die erforderlich ist, um ein Formular anzusehen (Zum Beispiel 1 = Admin, 2 = Hauptbenutzer, 3 = normaler Benutzer usw.)	1 (nur Administratoren können dieses Formular ansehen)
NewLevel	Die Benutzerstufe, die erforderlich ist, um neue Datensätze dem Formular hinzufügen zu können.	1 (nur Administratoren können neue Datensätze hinzufügen)
EditLevel	Die Benutzerstufe, die erforderlich ist, um Datensätze bearbeiten zu können.	1 (nur Administratoren können Datensätze bearbeiten)
Eraselevel	Die Benutzerstufe, die erforderlich ist um auf diesem Formular Datensätze löschen zu können.	1 (nur Administratoren können Datensätze löschen)
Favorites	Dieses Formular kann dem Favoriten-Menü hinzugefügt werden.	.T.
PrimaryKey	Der Primärschlüssel wird für die Verwaltung der Favoriten benötigt.	ID
FavorDescr	Beschreibung für den Eintrag im Favoriten-Menü.	
InetLevel	Zugriffsrecht auf AFP-Formulare	1 (nur Administratoren können AFP-Formulare anzeigen)

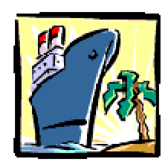
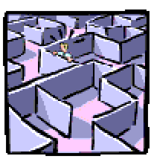
15.2. Systemeinstellungen im Optionen-Dialog

Im Optionen-Dialog können die Felder der Tabelle *Vfxsys.dbf* bearbeitet werden. Der Programmierer kann dieser Tabelle Felder mit globalen Einstellungen hinzufügen. Zur Laufzeit stehen die Werte aller Felder der Tabelle *Vfxsys.dbf* als Eigenschaften des global sichtbaren Objekts *goSystem* zur Verfügung.

15.3. Active Desktop

Der Active Desktop gibt den Anwendungen ein professionelles Startbild. Auf dem sonst leeren Bildschirm werden Bilder und Auswahlmöglichkeiten angeboten. Durch das Bewegen der Maus über die Bilder wird das zugehörige Menü unterhalb der Bilder angezeigt. In den Menüs befinden sich unterstrichene Menüpunkte, die ähnlich Hyperlinks im Internet Explorer, einfach angeklickt werden können und eine Aktion ausführen. In den meisten Fällen wird als Aktion ein Formular gestartet werden.

Die Klasse des Active Desktop befindet sich in der Klassenbibliothek *Appl.vcx* und kann nach den Wünschen des Entwicklers um beliebige Steuerelemente erweitert werden.

Simple

Parent	Parent form wich acts as parent form in a linked child scenario plus more...
Child	The same child form, just called directly, why not...
Item	Item table, shows the cTableForm class, very handy...
OneToMany	OneToMany form with parent -> child, almost a classic...
OneToMany2	OneToMany form item -> child, you are flexible, arent't you...
ParentTree	Parent Tree form shows the cTreeView class
OneToTree	Shows the cTreeViewOneToMany class

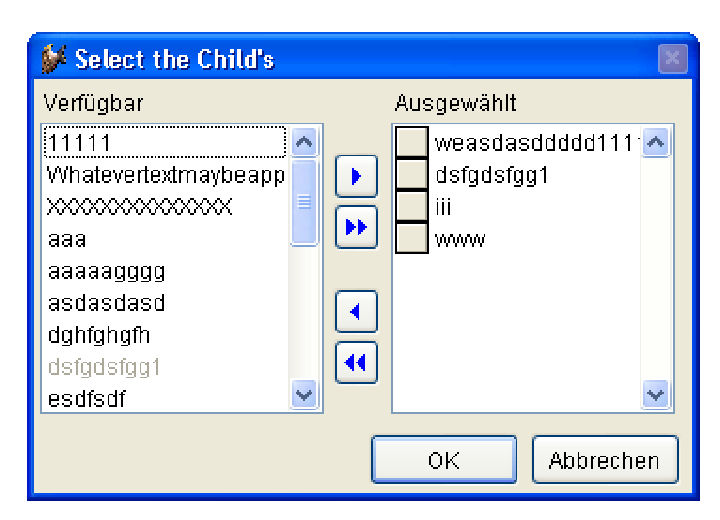
Der Active Desktop kann zusätzlich oder anstelle des Öffnen-Dialogs verwendet werden.

15.4. Weitere Funktionen

Über eine Formulareigenschaft (*lMore*) kann die Schaltfläche „weitere Funktionen" in der Standard-Symbolleiste aktiviert werden. Im *Click()*-Ereignis dieser Schaltfläche wird die *OnMore()*-Methode des aktiven Formulars aufgerufen. In dieser Methode steht bereits ein Template-Code, der leicht verändert werden kann. Hier werden in einem Array die Parameter für das VFXMore-Formular aufgerufen in dem in einem Dialog zwischen den zur Verfügung stehenden Funktionen ausgewählt werden kann. Z. B. können Child-Formulare gestartet werden.

15.5. Mover-Dialog

Der Mover-Dialog ist ein praktisches Werkzeug zur Auswahl von relativ wenigen Daten. Der VFX-Mover-Dialog bekommt als Parameter zwei Arrays übergeben. Das erste Array enthält zur Auswahl stehende Elemente. Diese Elemente werden in der linken Listbox angezeigt. Das zweite Array enthält die ausgewählten Elemente. Das zweite Array kann bei Aufruf des Mover-Dialogs leer sein. Der Anwender kann eine beliebige Anzahl von Elementen auswählen.



Hier ein Beispielcode für die praktische Anwendung des VFX-Mover-Dialog-Steuerelements:

```
LOCAL laSource[1,1], loMover

*--prepare the array of all available items
SELECT keygrp_id, keygrp_name FROM keygrp INTO ARRAY laSource

*--create the mover object based on the VFX Class CMoverDialog
loMover = CREATEOBJECT("CMoverDialog")
*--set the caption
loMover.Caption = CAP_KEYFIELDGEN
*--set the property which defines which column from the array get's displayed
loMover.cntMover.nColToView = 2
*--enable multiple selections
loMover.cntMover.lstSource.MultiSelect = .T.
*--pass the array of all available items
*  here you can also pass a second parameter if you want to define, which
*  elements from the array must appear as already selected
loMover.cntMover.SetData(@laSource)
*--show the mover dialog
loMover.Show()

*--Result: The Public Array _gaMoverList contains the selected items, use it
*  and release this Public Array after you have done.
```

Nach der Erstellung des Objektes *loMover* haben Sie die vollständige Kontrolle darüber und können alle gewünschten Eigenschaften und Methoden verändern.

ANMERKUNG: Um eine detaillierte technische Beschreibung der VFX-Klassenbibliotheken inklusive aller Eigenschaften und Methoden zu erhalten, lesen Sie bitte in der VFX Technischen Referenz nach.

15.6. OLE-Klassen

Es ist möglich Word, Excel, Outlook und Powerpoint per OLE aus VFX-Anwendungen anzusteuern. Die wichtigsten Funktionen stehen in Klassen zur Verfügung.

15.7. Debug-Modus

Durch Setzen einer Konstanten kann die Anwendung im Debug-Modus gestartet werden. Im Debug-Modus ist ein zusätzliches Menü sichtbar, mit dessen Hilfe jederzeit der Debugger gestartet werden kann. Außerdem kann durch einen Rechtsklick mit der Maus auf einem Formular der Debugger gestartet werden. Dabei wird auch das Set-Fenster geöffnet.

VFX benutzt eine Konstante in der Include-Datei *VFX.h*, die angibt ob die Anwendung im Debug-Modus ablaufen soll oder nicht. Standardmäßig sind die folgenden Codezeilen in der Datei *Vfxmain.prg*, um den Debug-Modus in Abhängigkeit von der Konstanten *_DEBUG_MODE* einzustellen:

```
#ifdef _DEBUG_MODE
  goProgram.DebugMode(.t.)
#endif
```

Wenn Sie nicht wollen, dass Ihre Anwendung im Debug-Modus ausgeführt wird, kommentieren Sie Zeile mit der *_DEBUG_MODE*-Konstanten aus. Die Konstante befindet sich in der Include-Datei *VFX.h*:

```
…
* #DEFINE _DEBUG_MODE         .T.
…
```

15.8. Delayed Instantiation

Die Ladezeit eines Formulars hängt im Wesentlichen von der Anzahl der Steuerelemente ab, die mit dem Formular geladen werden müssen. Nun sind aber in der Regel nicht alle Steuerelemente eines Formulars sofort sichtbar, wenn ein Formular gestartet wird. Wenn mit einem Seitenrahmen gearbeitet wird, sind zunächst nur die Steuerelemente einer Seite sichtbar. Die Steuerelemente der anderen, zunächst nicht sichtbaren Seiten, brauchten also gar nicht geladen werden. Erst wenn der Benutzer erstmals eine andere Seite aktiviert, müssen die auf dieser Seite befindlichen Steuerelemente nachgeladen werden.

Die Delayed Instantiation wird von VFX mit der sehr praktischen Funktion *addpagedelay()* unterstützt.

Um das Ziel zu erreichen müssen zunächst alle Steuerelemente einer Seite eines Pageframes in einem Container als Klasse gespeichert werden. Dafür markiert man im VFP Formular-Designer alle Steuerelemente der aktuellen Seite und wählt im Menü File den Punkt „Save As Class“. Die Klasse sollte in der Klassenbibliothek *Appl.vcx* gespeichert werden. Diese Klassenbibliothek steht dem Entwickler für eigene Klassen zur Verfügung. Beim Speichern als Klasse ergänzt VFP automatisch einen Container um die ausgewählten Steuerelemente. Der Name der Klasse sollte so gewählt werden, dass der Bezug zu dem Formular und der Seite des Pageframes leicht ersichtlich sind. Die als Klasse gespeicherten Steuerelemente können nun von dem Seitenrahmen gelöscht werden.

Um den Container zur Laufzeit des Formulars nachzuladen wird die Funktion *addpagedelay()* verwendet. Der Aufruf muss in das *Activate()*-Ereignis der jeweiligen Seite eingefügt werden und sieht so aus:

```
AddPageDelay(thisform, this, 'x', '<Name der Klasse>')
```

Es empfiehlt sich ein Formular zunächst ohne Delayed Instantiation zu entwickeln und zu testen. Wenn das Formular fast fertig ist, kann es auf Delayed Instantiation umgestellt werden. Zu beachten ist dabei, dass Referenzen auf einzelne Steuerelemente geändert werden müssen. Während vor der Umstellung auf Delayed Instantiation auf eine Textbox zum Beispiel so referenziert werden konnte:

```
Thisform.pgfPageframe.Page1.txtMeinetextbox
```

Sieht die Referenz nach Umstellung auf Delayed Instantiation so aus:

```
Thisform.pgfPageframe.Page1.x.txtMeinetextbox
```

Das *x* ist hierbei der Name des Containers, in dem sich die Steuerelemente der Seite befinden.

15.9. Wichtige VFX-Methoden

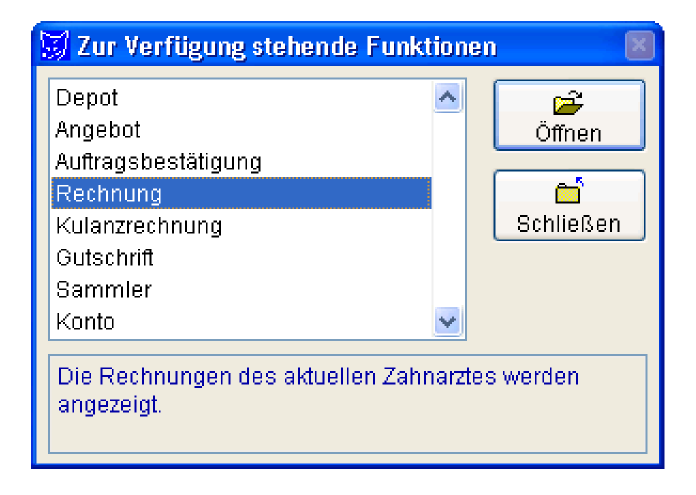
15.9.1. Formularmethoden

Valid

VFX bietet eine Valid-Methode auf Formularebene. Diese Methode wird immer aufgerufen, wenn die Daten des Formulars gespeichert werden sollen. Hier sollten also alle Validierungen untergebracht werden. Wenn aus dieser Methode der Wert .F. zurückgegeben wird, wird der Speichervorgang nicht fortgesetzt und das Formular bleibt im Bearbeitungsmodus. Durch Rückgabe von .T. werden die Daten gespeichert.

OnMore

Mithilfe dieser Methode ist es insbesondere möglich Child-Formulare aufzurufen. Ein fertiger Template-Code kann auf Wunsch vom VFX – Form Builder im Formular eingetragen werden. Je nach Anwendungsfall brauchen nur noch wenige Werte dieser Methode vom Entwickler angepasst werden.



Über die *OnMore()*-Methode wird zur Laufzeit ein Dialog angezeigt, in dem der Benutzer das aufzurufende Child-Formular auswählen kann.

Onpostinsert

Diese Methode wird unmittelbar nach dem Anfügen eines neuen Datensatzes aufgerufen, noch bevor der Benutzer die Möglichkeit zur Bearbeitung der Daten erhält.

Hier können also Standardvorgaben in den Feldern eingetragen werden. Diese Methode bietet sich auch an, um Primärschlüssel zu vergeben.

Onrecordmove

Jedes Mal, wenn der Satzzeiger bewegt wird, wird diese Methode aufgerufen. Hier können Werte angezeigt oder aktualisiert werden, die nicht aus der Datenbank stammen.

15.9.2. Methoden des Anwendungsobjekts

OnPreStart

In dieser Methode kann Code eingetragen werden, der vor Ausführung der Start-Methode ausgeführt werden soll.

OnPostStart

In dieser Methode kann Code eingetragen werden, der nach Ausführung der Start-Methode ausgeführt werden soll.

15.10. Primärschlüssel-Generierung

Es kann Tabellen geben, aus denen Sie den Primärschlüssel nicht den Benutzern zeigen wollen. Aber für ein korrektes Datenbankdesign wollen Sie einen Primärschlüssel verwenden. Für diese und ähnliche Situationen bietet VFX eine Funktion, die die Erstellung von Primärschlüsseln ermöglicht und in einer Mehrbenutzer-umgebung genauso funktioniert, wie in einer Client/Server-Umgebung.

Durch das modulare Design der VFX-Klassenhierarchie, haben Sie die Möglichkeit, nach dem Einfügen eines neuen Datensatzes einzugreifen. VFX bietet, neben vielen anderen Funktionen, eine Methode mit dem Namen *OnPostInsert(),* die in dem Moment ausgeführt wird, wenn ein neuer Datensatz gerade hinzugefügt wurde. Normalerweise bietet VFX für alle wichtigen Ereignisse Methoden, die automatisch vor, während und nach dem Ereignis ausgeführt werden. In diesem Fall, in dem ein neuer Datensatz hinzugefügt wird, gibt es die folgenden Methoden:

- *OnPreInsert()*
- *OnInsert()*
- *OnPostInsert()*

Außerdem gibt es eine Eigenschaft die angibt, ob der Benutzer einen neuen Datensatz aufnehmen kann. Diese Eigenschaft trägt den Namen *lCanInsert.*

ANMERKUNG: Für weitere Informationen lesen Sie bitte die VFX Technische Referenz.

Um einen Primärschlüssel zu erzeugen, könnten Sie in die *OnPostInsert()*-Methode Ihres Formulars etwa fol-genden Code einfügen. Hierdurch wird die Funktion *GetNewId()* aufgerufen. Der Parameter gibt die Tabelle an, für die der Schlüssel generiert wird.

```
DODEFAULT()
REPLACE comp_id WITH GetNewId('CUSTOMER') IN customer
```

Der Zähler für den generierten Schlüssel wird in der Tabelle *Vfxsysid.dbf* gespeichert.

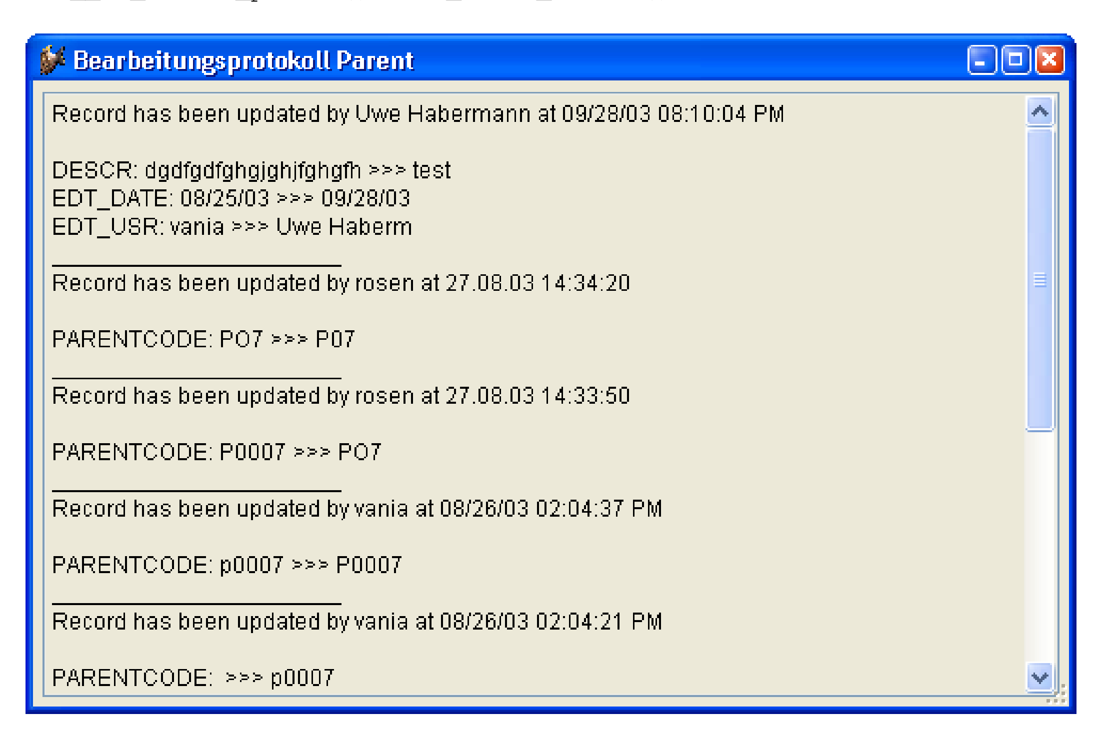
15.11. Bearbeitungsprotokoll

Das Bearbeitungsprotokoll (Audit-Trail) protokolliert Änderungen von Daten. VFX verwendet Trigger um die Änderung von Daten zu ermitteln. Die Trigger-Funktionen werden bei allen zu überwachenden Tabellen eingetragen.

- *_audit_insert()* protokolliert die Erfassung neuer Datensätze
- *_audit_update()* protokolliert alle Änderungen
- *_audit_delete()* protokolliert das Löschen von Datensätzen

Ein Audit-Trigger kann mit einem RI-Trigger mit einem logischen „und" verknüpft werden:

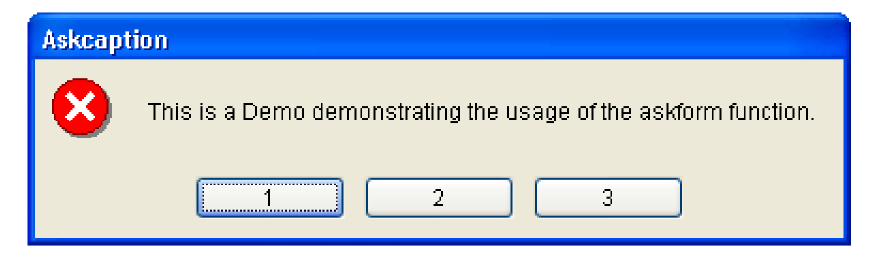
```
__ri_delete_parent() AND _audit_delete()
```



Über eine Schaltfläche in der Standard-Symbolleiste kann zum aktuell angezeigten Datensatz das Änderungsprotokoll angesehen werden.

15.12. Askform

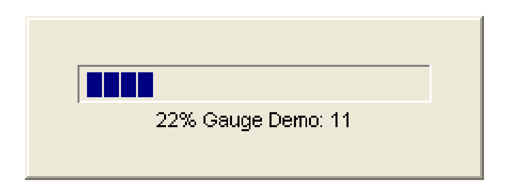
Die Askform entspricht in etwa einer Messagebox, hat jedoch eine erweiterte Funktionalität. Die Beschriftungen der maximal drei Schaltflächen können als Parameter übergeben werden. Außerdem ist es möglich ein Timeout für die Messagebox festzulegen. Bei erreichen des Timeouts ohne Benutzeraktion wird ein Rückgabewert geliefert, der dem Drücken der Standard-Schaltfläche entspricht.



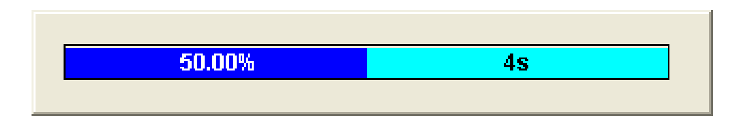
Ein Beispiel zur Verwendung der Funktion *Askform()* befindet sich im Formular *Parent.scx* aus der Demoanwendung VFX110Test.

15.13. Fortschrittsanzeige

VFX bietet zwei Möglichkeiten den Fortschritt von lange andauernden Vorgängen zu verdeutlichen.
Die einfache Variante, realisiert mit der Formularklasse *CGaugeWin*, zeigt einen Balken zur Anzeige des Fortschritts an.



Mit dem Formular *Vfxmtr.scx* kann eine Fortschrittsanzeige mit Anzeige der Restzeit dargestellt werden.

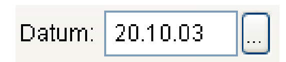


Beispiele für die Verwendung beider Fortschrittsanzeigen befinden sich im Formular *Parent.scx* der Demoanwendung VFX110Test.

15.14. Datumsauswahl

15.14.1. Die Klasse CPickDate

Dis Klasse *CPickDate* enthält eine Textbox zur Eingabe eines Datums sowie eine Schaltfläche zum Aufruf eines Kalenders.



In der Textbox stehen die folgenden Hotkeys zur Auswahl eines Datums zur Verfügung:

+	Nächster Tag
-	Vorheriger Tag
H, h	Heute
B, b	Der erste Tag (Beginn) des angezeigten Monats
L, l	Der letzte Tag des angezeigten Monats
A, a	Neujahr
E, e	Sylvester
V, v	Vorheriger Monat
N, n	Nächster Monat

Für den Kalender wird das ActiveX-Steuerelement Microsoft MonthView verwendet. Bei der Erstellung eines Setups muss dieses ActiveX-Steuerelement (*Mscomct2.ocx*) mit in das Setup einbezogen werden. VFP 9 stellt hierfür ein Merge Module bereit.



15.14.2. Die Klasse CDatetime

Zusätzlich steht die Klasse *CDatetime* zur Eingabe von *Datetime*-Werten zur Verfügung.



In dieser Klasse ist zur Eingabe des Datums ein *CPickDate*-Steuerelement enthalten. Es stehen alle Funktionen des *CPickDate*-Steuerlements wie zum Beispiel der Kalender oder die Hotkeys zur Verfügung.

Um eine Zeiteingabe im 24-Stunden-Format zu ermöglichen muss SET HOURS TO 24 eingestellt sein. Diese Einstellung kann global für alle Formulare in der Funktion *formsetup()* in *Applfunc.prg* gemacht werden.

Die Controlsource der Klasse *CDatetime* wird in der Eigenschaft *ccontrolsource* eingestellt. Die Controlsource muss vom Typ *Datetime* sein.

15.15. Auswahl von Berichten

Wenn zu einem Formular verschiedene Berichte gedruckt werden sollen, bietet die Klasse *CRSelection* einen geeigneten Auswahldialog. Die zur Verfügung stehenden Berichte werden aus Tabellen gelesen. Es kann zwischen Berichten unterschieden werden, die für alle Benutzer sichtbar sind und Berichten, die nur für einzelne Benutzer sichtbar sind.

Ein Beispiel zur Anwendung findet sich im Formular *Reports.scx* in der Demoanwendung VFX110Test.

15.16. Die Microsoft Agents

Die Agents sind nette Charaktere, die die Benutzung von VFX-Anwendungen auflockern.

In VFX110Test zeigt das Formular *Agent.scx* einfache Beispiele für die Verwendungsmöglichkeiten.

15.17. Die VFX-Ressourcentabelle

VFX-Anwendungen verwenden eine Ressourcentabelle, in der je Benutzer Informationen über alle Formulare, die der Benutzer bereits einmal verwendet hat, gespeichert sind. Hierbei werden nicht nur die Positionen der Formulare, sondern auch Layoutänderungen an Grids inklusive der Sortierfolgen gespeichert.

VFX-Anwendungen verwenden nicht die Visual FoxPro Ressourcentabelle *Foxuser.dbf*, stattdessen verwenden Sie ausschließlich die freie VFX-Ressourcentabelle *Vfxres.dbf*.

Hier die Einstellungen, die in der VFX-Ressourcentabelle je Benutzer gespeichert werden.

Einstellung	**Beschreibung**	**Bemerkung**
Position und Größe von Formularen	Der Benutzer sieht die Formulare bei erneutem Öffnen genau so, wie er sie zuletzt verlassen hat.	Individuelle Formulareinstellungen **Hinweis:** Bezieht sich auch auf Auswahllisten!
Alle vorgenommenen Layoutänderungen an Grids	Der Benutzer sieht die Grids genau so, wie er sie verlassen hat. Sowohl Spaltenbreiten als auch Anordnung (auch wenn es sich hierbei um berechnete Felder handelt).	Individuelle Grid-Einstellungen **Hinweis:** Bezieht sich auch auf Auswahllisten sowie 1:n-Formulare mit mehreren Child-Grid!
Aktuelle Sortierung der Datenbearbeitungsformulare sowie der Auswahllisten	Die letzte Sortierfolge wird automatisch wiederhergestellt. Unabhängig davon, ob ein Indexschlüssel vorhanden ist oder nicht. VFX erstellt temporäre IDX-Dateien für nicht vorhandene Schlüssel.	VFX erstellt automatisch benötigte IDX-Dateien im temporären Windows-Ordner und löscht diese wieder beim Verlassen des Formulars. **Hinweis:** Bezieht sich auch auf Auswahllisten!
Position und Status von Symbolleisten	Falls Sie eine Symbolleiste an ein Formular anbinden, so wird diese in demselben Status präsentiert, wie sie beim letzten Arbeiten mit diesem Formular verlassen wurden.	
Unterdrückung von Symbolleisten	Falls der Benutzer die formularspezifische Symbolleiste geschlossen hat, so wird diese bei erneutem Öffnen dieses Formulars nicht mehr geöffnet. Um die Symbolleiste erneut zu aktivieren, muss der Symbolleisten-Dialog aus dem Menü *Ansicht* geöffnet werden und die entsprechende Symbolleiste geöffnet werden.	

Sie können Ihre Ressourcendaten in der Benutzerverwaltung löschen.

15.18. Include-Dateien

Die Include-Dateien spielen bei VFX eine wichtige Rolle. Es lohnt sich deshalb, die vorhandenen Include-Dateien etwas näher anzusehen:

Include-Datei	**Verwendung**	**Sprach-abhängig?**	**Inhalt/Beschreibung**
FoxPro.h	VFX.H	Nein	Standard-FoxPro-Definitionen.

FoxPro_Reporting.h	VFX.H	Nein	Konstanten für Druckfunktionen von VFP.
ReportListeners.h	VFX.H	Nein	Konstanten für die ReportListener Klasse von VFP.
ReportListeners_Loc.h	VFX.H	Ja	Zu lokalisierende Texte für den ReportListener von VFP.
UserDef.h	VFX.H	Nein	Sprachunabhängige Konstanten, die in Ihrer Anwendung verwendet werden.
UserMsg.h	VFX.H	Ja	Sprachabhängige Meldungstexte, die Sie in Ihrer eigenen Anwendung verwenden. Die Datei wird von dem VFX – Message Editor erzeugt, wenn Sie die Option MESSAGE wählen.
UserTxt.h	VFX.H	Ja	Sprachabhängige Texte und Tooltip-Texte, die Sie in Ihrer eigenen Anwendung verwenden. Die Datei wird von dem VFX – Message Editor erzeugt, wenn Sie die Option OTHER wählen.
VFX.h	VFXMAIN.PRG	Nein	Definiert die Konstanten _DEBUG_MODE, *LANG*SETUP, _DBCX und schließt andere Include-Dateien ein.
Vfxdef.h	VFX.H	Ja	Definiert die ID_LANGUAGE-Konstante und andere Konstanten.
VfxGlobal.h	VFX.H	Ja	Konstanten für Felder aus der Benutzerverwaltung und aus dem Optionendialog. Diese Datei wird aus Kompatibilitätsgründen zu früheren VFX-Versionen benötigt.
Vfxmsg.h	VFX.H	Ja	Sprachabhängige Meldungstexte, die in VFX-Anwendungen verwendet werden.
Vfxoffce.h	VFX.H	Nein	In den Office-Klassen Word, Excel und Outlook verwendet.
VfxToolbox.h	VFX.H	Ja	Enthält Konstanten für die VFP Toolbox.
VfxTxt.h	VFX.H	Ja	Sprachabhängige Texte und Tooltip-Texte, die in VFX-Anwendungen verwendet werden.
_FrxCursor.h	VFX.H	Ja	

Der VFX Anwendungs-Assistent generiert die meisten Konstanten automatisch, wenn Sie ein neues Projekt generieren. Wenn Sie den Debug-Modus wechseln wollen, müssen Sie Änderungen in der Include-Datei *VFX.h* machen.

Um Visual FoxPro zu einem Neukompilieren zu veranlassen, müssen Sie eine Änderung in der oder den Datei(en) vornehmen, die die Include-Dateien einschließen. Der Befehl *clear program* im Befehlsfenster löscht alle kompilierten Programme im Hauptspeicher. Zusätzlich sollten die Dateien *Program*.fxp* und *Menu*.fxp* unterhalb des Projektordners gelöscht werden. Sie sollten die Datei *VFX.h* in Ihre Formulare einschließen, wenn Sie Konstanten in Ihren Formularen verwenden.

15.19. OLE drag & drop

In VFX-Anwendungen steht OLE drag & drop auf drei verschiedene Arten zur Verfügung. Standardmäßig ist OLE drag & drop in Grids eingeschaltet. Der gesamte Inhalt eines Grid kann mit einem Mausklick zum Beispiel nach Excel kopiert werden.

Auf Wunsch können auch die Inhalte einzelner Steuerelemente per OLE drag & drop verschoben werden. Diese Eigenschaft ist standardmäßig ausgeschaltet und kann im VFX – Application Builder über die Eigenschaft nOLEenableDrag des Anwendungsobjekts eingeschaltet werden.

```
nOLEenableDrag=1  && 0 use form setting (default), 1 enable, 2 disable
```

Weiterhin ist es möglich die Daten aller Steuerelemente einer Seite eines Seitenrahmens in eine andere OLE drag & drop-fähige Anwendung zu kopieren. Auch diese Eigenschaft ist standardmäßig ausgeschaltet und kann bei Bedarf im VFX – Application Builder über die Eigenschaft *nPageOLEdragdrop* des Anwendungsobjekts eingeschaltet werden.

```
nPageOLEdragdrop=1   && 0 use form setting (default), 1 enable, 2
disable
```

15.20. Hooks

VFX bietet bei allen wichtigen Methoden Eingriffsmöglichkeiten über Hooks. Als Beispiel schauen wir die *OnInsert()*-Methode eines Formulars an. Die *OnInsert()*-Methode wird aufgerufen, wenn ein neuer Datensatz

angefügt werden soll. Dabei wird zunächst die Methode *OnPreInsert()* aufgerufen. Nur wenn diese Methode .T. als Rückgabewert liefert, wird ein Datensatz angefügt. Nach dem Anfügen des Datensatzes wird die *OnPostInsert()*-Methode aufgerufen. Hier können z. B. mit dem Replace-Befehl Daten in den neuen Datensatz eingetragen werden. Wenn die *OnPostInsert()*-Methode .F. zurückliefert, wird ein *Tablerevert()* durchgeführt und der neue Datensatz damit sofort wieder gelöscht.

Eine elegante Möglichkeit in den Funktionsablauf von VFX-Methoden einzugreifen, ohne die Klassen verändern zu müssen, ist der Einsatz von Hooks.

In den meisten VFX-Methoden ist ein Eventhook eingebaut. Wenn die Eventhooks aktiviert sind, wird in jedem Eventhook die Funktion Eventhook-Handler aufgerufen. Als Parameter werden dieser Funktion der Name der aufrufenden Methode, eine Referenz auf das aktuelle Objekt und eine Referenz auf das aktuelle Formular übergeben. Über eine Case-Konstruktion kann dann individueller Code ausgeführt werden. Hierdurch kann an praktisch jeder Stelle in den Funktionsablauf von VFX eingegriffen werden.

Das Konzept der Hooks wurde in VFX 11.0 erweitert. Bisher war es möglich durch einen Hook innerhalb einer VFX-Methode einen eigenen Codeblock auszuführen. Über den Rückgabewert des Hooks konnte man steuern, ob der noch folgende VFX-Code in der Methode weiter ausgeführt werden sollte oder nicht. Der Rückgabewert, den die VFX-Methode dabei lieferte, konnte nicht beeinflusst werden und war in VFX fest vorgegeben.

Mit den erweiterten Hooks in VFX 11.0 kann nun zusätzlich der Rückgabewert der Methode vom Hook gesteuert werden.

Hooks sind in der Datei *Vfxhook.prg* gespeichert. Die Verwendung von Hooks kann im VFX – Application Builder mit der Eigenschaft *nenablehook* = 1 eingeschaltet werden. *Nenablehook* ist eine Eigenschaft des Anwendungsobjekts.

Im folgenden Beispiel wird bei allen Steuerelementen, die disabled sind, die Schriftfarbe schwarz eingestellt.

```
function EventHookHandler(tcEvent, toObject, toForm)
   local lContinue
   lContinue = .T.
   DO CASE
   CASE UPPER(tcEvent)=="INIT"
      IF PEMSTATUS(toObject,"disabledforecolor",5)
         toObject.disabledforecolor=;
            eval(left(rgbscheme(1,2),at(",",rgbscheme(1,2),3)-1)+")")
         IF PEMSTATUS(toObject,"disabledbackcolor",5)
            toObject.disabledbackcolor=;
               eval("rgb("+substr(rgbscheme(1,2),;
               at(",",rgbscheme(1,2),3)+1))
         ENDIF
      ENDIF
   ENDCASE
   return lContinue
endfunc
```

15.21. Geschäftsgrafiken

Statistische Auswertungen in endlosen Listen sind schwer zu lesen und zu analysieren. Der bessere Weg zur Veranschaulichung von Geschäftsdaten sind grafische, farbige Präsentationen. Die neue Klasse *CBusinessGraph* gibt dem VFX-Entwickler die Möglichkeit Anwendungsdaten mit nur wenigen Minuten Programmierarbeit in Grafiken anzuzeigen und zu drucken.

Zur Anzeige der Grafiken wird das ActiveX-Steuerelement *MSChart* eingesetzt. Die anzuzeigenden Daten können aus einem beliebigen Cursor kommen. Jede Spalte des Cursors entspricht einer Koordinate in der Grafik. Eins der Felder kann Bezeichnungstexte enthalten. Wenn kein Feld mit Bezeichnungstexten angegeben wird, werden alle Felder des Cursors zur Datenanzeige verwendet. Felder zur Datenanzeige müssen einen numerischen Datentyp haben. Zusätzlich können Texte für die Legende der Grafik angegeben werden.

Eigenschaften

cAliasName – Aliasname des Cursors, der die Daten enthält.

cGraphTitle – Titel der Grafik.

cLabelField – Name des Feldes, das die Bezeichnungstexte enthält.

cLegendTitles – Eine komma-separierte Liste mit der Legende.

lShowLegend – Wenn der Wert dieser Eigenschaft auf .T. eingestellt wird, wird neben der Grafik eine Legende angezeigt.

nGraphType – Anzeigetyp der Grafik:

0 - 3D Balken (Säule)
1 - 2D Balken/Piktogramm
2 - 3D Linie (Band)
3 - 2D Linie
4 - 3D Fläche
5 - 2D Fläche
6 - 3D Schritt
7 - 2D Schritt
8 - 3D Kombination
14 - 2D Kreis
16 - 2D X Y (Punkt)

Methoden

DrawGraph – Erstellen der Grafik anhand der zur Verfügung stehenden Daten und der zuvor eingestellten Eigenschaften. Alle Bezeichnungen und die Legende werden aktualisiert.

OnPrint – Druckt die aktuelle Grafik mit der Berichtsvorlage *Hardcopy.frx*.

15.21.1. Beispiel

Ein Programmteil einer Anwendung erstellt den folgenden Cursor. Daraus soll eine Geschäftsgrafik erstellt werden.

Yearlysales

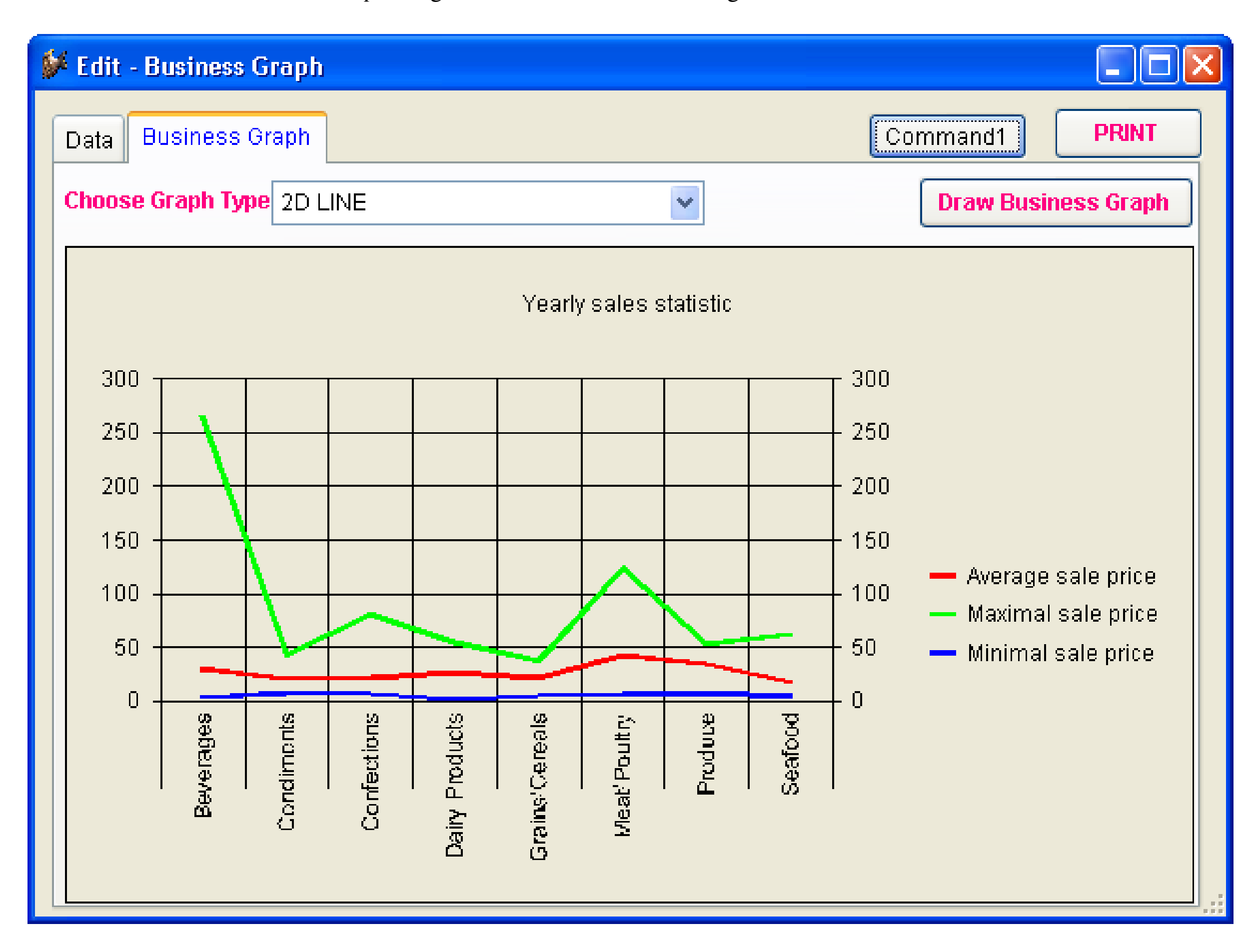
Categoryname	Averageprice	Maxprice	Minprice
Beverages	29.9892	263.5000	3.6000
Condiments	21.3968	43.9000	8.0000
Confections	22.3396	81.0000	7.3000
Dairy Products	27.4055	55.0000	2.0000
Grains/Cereals	22.0311	38.0000	5.6000
Meat/Poultry	42.3548	123.7900	5.9000
Produce	35.1579	53.0000	8.0000
Seafood	18.3691	62.5000	4.8000

Die Klasse *CBusinessGraph* kann auf ein beliebiges Formular gezogen werden. Die folgenden Einstellungen werden bei dem Objekt gemacht:

```
.cAliasName = "YearlySales"
.cGraphTitle = "Yearly sales statistic"
.cLabelField = "CategoryName"
.cLegendTitles = "Average sale price, Maximal sale price, Minimal sale
price"
```

Der Eigenschaft *cLabelField* wird der Name der Spalte für die Bezeichnungen zugewiesen. Der Eigenschaft *cLegendTitles* wird eine Aufzählung der Texte für die Legende zugewiesen. Die Reihenfolge der Texte muss der Reihenfolge der Spalten im Cursor entsprechen.

Wenn nun die Methode *DrawGraph* ausgeführt wird, erscheint die folgende Grafik.



15.22. Symbolleisten

15.22.1. Benutzen Sie die gewünschte Standard-Symbolleiste

Es ist vernünftig, für die Bedürfnisse Ihrer Anwendung eine eigene Klassenbibliothek anzulegen. Wir haben eine Klassenbibliothek mit dem Namen *Appl.vcx* für Sie vorbereitet. In dieser Klassenbibliothek befinden sich unter anderem die beiden Klassen für die Symbolleisten

CAppToolBar und *CAppNavBar*.

Die Erste ist die Standard-Symbolleiste und die Zweite ist eine Symbolleiste, die Sie verwenden können, wenn Sie Navigations- und andere Schaltflächen nicht auf Ihren Formularen haben wollen.

CAppToolBar:



CAppToolBar wird benutzt, wenn die Schaltflächen zur Navigation und zur Bearbeitung auf Ihren Formularen sind.

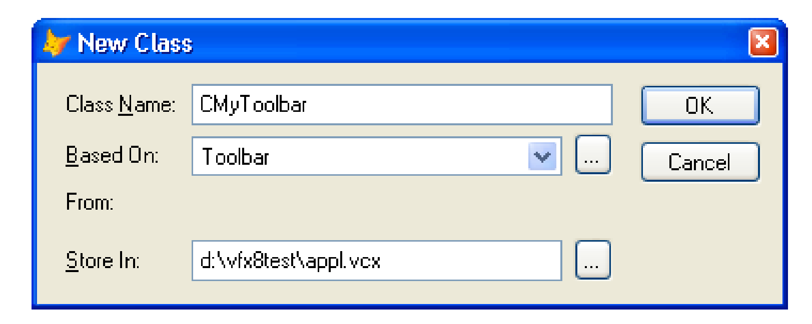
CAppNavBar:

CAppNavBar wird benutzt, wenn die Schaltflächen zur Navigation und zur Bearbeitung nicht auf Ihren Formularen sind.

Um zwischen diesen beiden Symbolleisten zu wechseln, brauchen Sie nur die Eigenschaft *CMainToolbar* mit dem VFX – Application Builder ändern.

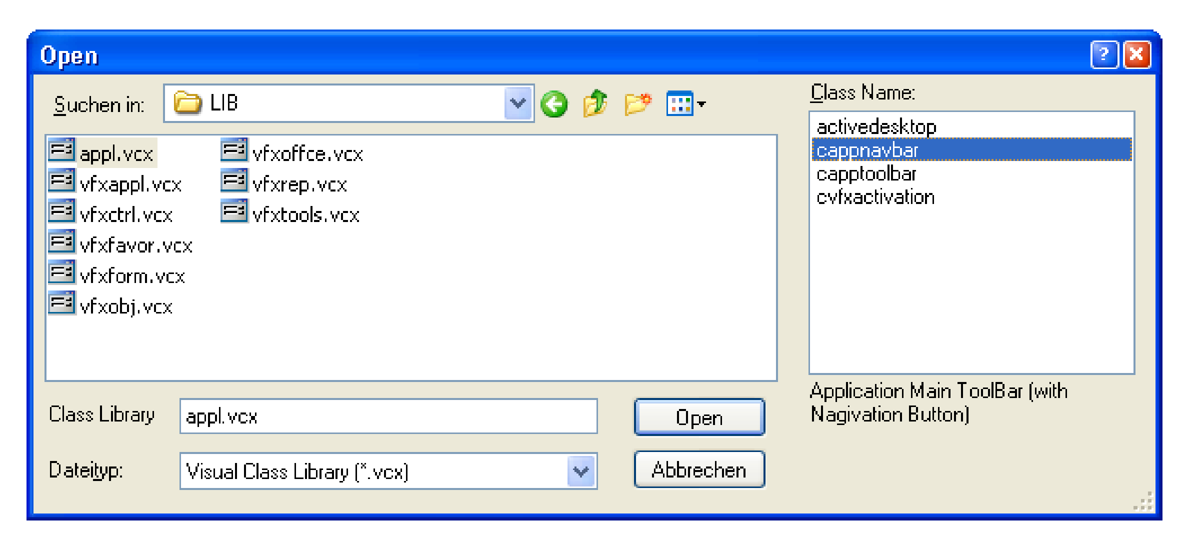
Sie können die *CAppToolBar*- oder die *CAppNavBar*-Symbolleistenklassen für die meisten Office-kompatiblen Anwendungen benutzen. Aber selbstverständlich können Sie auch andere Symbolleisten verwenden. Sie müssen nur eine neue Klasse erstellen, die von der *CToolbar*-Klasse oder auch von der *CAppToolBar*- oder der *CAppNavBar*-Klasse vererbt wird.

Wählen Sie *Neu,* wenn Sie sich auf der Klassenseite des Projekt-Managers befinden. Es wird folgendes Dialogfenster angezeigt:



Class Name: Geben Sie den Namen der neuen Klasse ein. Wir nennen sie hier *CMyToolbar*.

Based On: Drücken Sie auf die Schaltfläche mit den drei Punkten und das folgende Dialogfenster wird geöffnet. Wählen Sie die Klasse *CAppToolBar* (oder *CAppNavBar*) aus der VFX-Klassenbibliothek *Appl.vcx.*



From: Die Referenz auf die VFX-Klassenbibliothek mit dem Namen *Appl.vcx* wird automatisch angezeigt.

Store In: Wenn Ihre anwendungsspezifische Klassenbibliothek noch nicht existiert, geben Sie den vollständigen Pfadnamen an. Andernfalls wählen Sie Ihre Klassenbibliothek mit der Schaltfläche mit den drei Punkten (Dialog zur Dateiauswahl).

Jetzt müssen Sie Ihre Symbolleistenklasse anpassen. Sie machen dies mit dem Klassen-Designer.

Eine Schaltfläche einfügen

Visual Extend bietet vordefinierte Schaltflächen für die einfache Erstellung von Symbolleisten. Ziehen Sie die Klasse *CToolbarButton* aus der VFX-Klassenbibliothek *Vfxctrl.vcx* auf Ihre Symbolleiste und passen Sie die folgenden Eigenschaften und Methoden an Ihre Bedürfnisse an:

Click Event: Tragen Sie die Befehle ein, die immer dann ausgeführt werden sollen, wenn der Benutzer auf diese Schaltfläche drückt. Wenn Sie beispielsweise das Formular *Customer* öffnen wollen, geben Sie folgenden Code

```
goProgram.RunForm("CUSTOMER")
```

in das *Click()*-Ereignis ein.

Picture: Wählen Sie eine *Bmp*- oder *Ico*-Datei aus, die als Beschriftung Ihrer Schaltfläche angezeigt wird.

Fügen Sie den folgenden Code in das *Refresh()*-Ereignis jeder Schaltfläche oder Ihrer Symbolleiste ein. Sie stellen damit sicher, dass die Schaltflächen immer richtig angezeigt werden. Wenn Sie ein modales Formular öffnen, wird VFX die Schaltflächen in den Symbolleisten deaktivieren. Sie können mit folgendem Code sicherstellen, dass die Schaltflächen wieder richtig aktiviert werden:

```
this.enabled = this.parent.cmdopen.enabled
```

Mit diesem Code wird die Schaltfläche der Symbolleiste automatisch mit dem Anzeigeverhalten der Schaltfläche *Öffnen* synchronisiert.

Einen Zwischenraum einfügen

Fangen Sie mit einem Zwischenraum an, um die erste anwendungsspezifische Schaltfläche von der letzten Schaltfläche der Standard-Symbolleiste zu trennen.



Benutzen Sie dieses Symbol aus der Visual FoxPro Symbolleiste für Formular-Steuerelemente und ziehen Sie es auf Ihre Symbolleiste wo es benötigt wird.

15.22.2. Hinzufügen einer Symbolleiste zu einem Formular

Sehr anwenderfreundlich ist die Möglichkeit einem Formular eine Symbolleiste hinzuzufügen. Die Symbolleisten sollten auf der Klasse *CTtoolbar* basieren und in der Klassenbibliothek *Appl.vcx* gespeichert werden.

Der Name der Symbolleiste wird in der Eigenschaft *CToolbarClass* des Formulars eingetragen. VFX instanziiert die Symbolleiste zusammen mit dem Formular. VFX zeigt die Symbolleiste automatisch an, wenn das Formular aktiv ist und versteckt sie wieder, wenn ein anderes Formular aktiv wird. Selbstverständlich werden der Status und die Position der Symbolleiste benutzerspezifisch gespeichert.

Im *Click()*-Ereignis der Symbolleisten-Schaltflächen wird sinnvollerweise eine Methode des aktiven Formulars aufgerufen. Z. B.:

```
_screen.activeform.meinemethode()
```

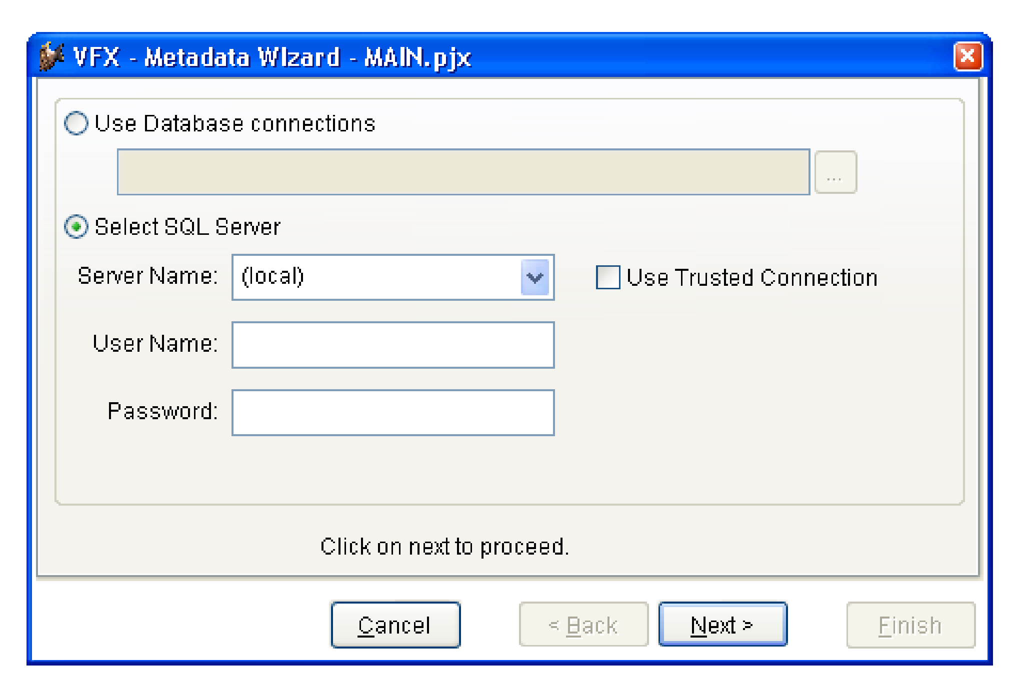
Um zum Beispiel ein Child-Formular über eine Schaltfläche in einer Symbolleiste zu öffnen, fügen wir der Symbolleiste eine Schaltfläche basierend auf der Klasse *CToolbarClass* hinzu. In das *Click()*-Ereignis der Schaltfläche schreiben wir

```
_screen.activeform.onmore(1)
```

Das ist alles. Da VFX sicherstellt, dass die Symbolleiste nur dann sichtbar ist, wenn das dazugehörige Formular aktiv ist, können wir sicher sein, dass *_screen.activeform* existiert. Von diesem Formular wird die *OnMore()*-Methode aufgerufen und bekommt als Parameter eine *1* übergeben. Damit wird das Formular aufgerufen, das im ersten Array-Element der *OnMore()*-Methode angegeben ist, ohne dass der *OnMore()*-Dialog angezeigt wird.

15.23. Die Klasse CWizard

Die Klasse *CWizard* ermöglicht die Erstellung von Assistenten. Der Anwender wird Schritt für Schritt durch die Bearbeitung geführt. Ein gutes Beispiel für die Verwendung der Klasse *CWizard* ist in den VFX-Wizards selbst enthalten. Der VFX – Metadata Wizard basiert auf der Klasse *CWizard*.



15.24. Die Klasse CDownload

Diese Klasse ermöglicht das Herunterladen von Dateien aus dem Internet. Bei Bedarf können die heruntergeladenen Dateien ausgeführt werden und es können weitere Aktionen ausgeführt werden. Insbesondere ist hierdurch die Installation von Programmen aus dem Internet möglich.

Durch die Verwendung von Makros und die *Execmacro*-Funktion von VFP kann diese Klasse sehr vielseitig eingesetzt werden.

Makros sind Zeichenketten, die eine Folge von Befehlen aus der Makrosprache enthalten. Eigene Makros können erstellt werden. Ein Beispiel ist in der Tabelle *Vfxsys.dbf* im Feld *Install_GS* zu finden. Mit diesem Makro wird das Programm Ghostscript aus dem Internet heruntergeladen und installiert.

Diese Klasse verwendet die in der Eigenschaft *goProgram.cConnectionCheckURL* gespeicherte Internetseite um zu überprüfen, ob eine Internetverbindung besteht. Bei Bedarf wird eine Verbindung automatisch hergestellt. Wenn im DFÜ-Netzwerk keine Verbindung eingetragen ist, wird ein neuer Eintrag hinzugefügt. Die Verbindungsinformationen kann der Entwickler in den Eigenschaften vorgeben. Der Anwender kann die Telefonnummer, den Benutzernamen und das Kennwort für die neue Verbindung bei Bedarf in einem Dialog ändern.

Eigenschaften

LastErrorNo – Diese Eigenschaft enthält die Nummer des letzten Fehlers (falls ein Fehler aufgetreten ist). Damit kann die Ursache des letzten Fehlers ermittelt werden.

LastErrorTest – Wenn ein Fehler aufgetreten ist, ist in dieser Eigenschaft der Text der Fehlermeldung zu finden.

Methoden

ExecMacro (vcMacro, lnNoRun)

vcMacro – Skript der Makrosprache, das ausgeführt werden soll.

lnNoRun – Wenn diese Eigenschaft auf .T. gesetzt wird, wird die heruntergeladene Datei nicht ausgeführt.

15.24.1. Befehle der Makrosprache

„D:“ URL

Unter dieser Internetadresse ist die herunterzuladende Datei zu finden. Dieser Befehl führt die Datei nach dem erfolgreichen Herunterladen aus, wenn die Eigenschaft *lnNoRun* auf .F. gesetzt ist.

„C:“ nTimeOut; lPartial; lTopLevelForm; lResultOnError; SearchedString

Wartet bis das Fenster mit dem Titel *SearchedString* erscheint.

nTimeOut – Timeout in Sekunden. Wenn das erwartete Formular nicht innerhalb dieser Zeitspanne erscheint, wird ein Timeout-Fehler erzeugt.

lPartial – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, reicht es, wenn der übergebene Titel einem Teil des Fensternamens entspricht. Wenn diese Eigenschaft auf .F. gesetzt ist, muss der übergebene Titel exakt dem Namen des Fensters entsprechen.

lTopLevelForm – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, wird der Fenstername nur in Top-Level-Fenstern gesucht.

lResultOnError – Mit dieser Eigenschaft wird das Verhalten des Skripts gesteuert, falls das Fenster nicht innerhalb der vorgegebenen Zeitspanne gefunden wurde. Wenn das Fenster für die weitere Ausführung des Skripts zwingend erforderlich ist, muss nach Ablauf der vorgegebenen Zeitspanne die Ausführung des Skripts abgebrochen werden. In diesem Fall muss der Wert von *lResultOnError* auf .F. gesetzt werden. Wenn die Ausführung des Skripts unabhängig vom Vorhandensein des Fensters nach der vorgegebenen Zeitspanne fortgesetzt werden soll, muss *lResultOnError* auf .T. gesetzt werden.

SearchedString –Bezeichnung, die in einem Fensternamen gesucht wird.

„W:“ nTimeOut; lPartial; lTopLevelForm; lResultByError; SearchedString

Es wird gewartet bis das Fenster, das die angegebene Zeichenkette im Titel enthält, geschlossen ist.

nTimeOut – Timeout in Sekunden. Wenn das erwartete Fenster innerhalb dieser Zeitspanne nicht geschlossen ist, wird ein Timeout-Fehler ausgelöst.

lPartial – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, reicht es, wenn der übergebene Titel einem Teil des Fensternamens entspricht. Wenn diese Eigenschaft auf .F. gesetzt ist, muss der übergebene Titel exakt dem Namen des Fensters entsprechen.

lTopLevelForm – Wenn der Wert dieser Eigenschaft auf .T. gesetzt ist, wird der Fenstername nur in Top-Level-Fenstern gesucht.

lResultOnError – Mit dieser Eigenschaft wird das Verhalten des Skripts gesteuert, falls das Fenster nicht innerhalb der vorgegebenen Zeitspanne gefunden wurde. Wenn das Fenster für die weitere Ausführung des Skripts zwingend erforderlich ist, muss nach Ablauf der vorgegebenen Zeitspanne die Ausführung des Skripts abgebrochen werden. In diesem Fall muss der Wert von *lResultOnError* auf .F. gesetzt werden. Wenn die Ausführung des Skripts unabhängig vom Vorhandensein des Fensters nach der vorgegebenen Zeitspanne fortgesetzt werden soll, muss *lResultOnError* auf .T. gesetzt werden.

SearchedString – Eine Zeichenkette nach der im Titel eines Fensters gesucht wird.

„X:“

Schließt das Top-Level-Fenster. Mit dem „*C:*“-Befehl muss zuvor sichergestellt werden, dass das gewünschte Fenster sichtbar ist.

„K:“ nKeyCode1; nKeyCode2; …

Die aufgeführten Tastenschlüssel werden in den Windows-Tastaturpuffer übertragen.

„U:“ URL

Von dieser Internetadresse wird das Herunterladen ausgeführt. Die heruntergeladene Datei wird unabhängig vom Wert der Eigenschaft *lnNoRun* nicht ausgeführt.

15.24.2. Beispiel

Beschreibung der Installation von Ghostscript:

D: ftp://mirror.cs.wisc.edu/pub/mirrors/ghost/AFPL/gs811/gs811w32.exe

Lädt die Datei gs811w32.exe aus dem Internet herunter und führt sie anschließend aus.

C: 30; .F.; .F.; .F.; WinZip Self-Extractor - gs811w32.exe

Wartet bis das Fenster mit dem Titel „WinZip Self-Extractor - gs811w32.exe“ erscheint.

K: 43

Sendet den Tastenschlüssel „Eingabetaste“ an das aktive Fenster. Dadurch wird das Entpacken der Dateien ausgelöst.

C: 60; .F.; .F.; .F.; AFPL Ghostscript Setup

Wartet bis das Fenster mit dem Titel „AFPL Ghostscript Setup“ erscheint.

K: 43

Sendet den Tastenschlüssel „Eingabetaste“ an das aktive Fenster. Dadurch wird die Installation von Ghostscript gestartet.

W: 240; .F.; .F.; .F.; AFPL Ghostscript Setup Log

Wartet solange das Fenster „AFPL Ghostscript Setup Log“ geöffnet ist. Dieses Fenster zeigt den Fortschritt der Installation an und die Skriptausführung muss warten, bis dieser Vorgang beendet ist.

C: 30; .T.; .T.; .T.; Ghostscript

Wartet bis das Fenster mit dem Titel „Ghostscript“ erscheint. Dieses Fenster zeigt die Nachricht an, dass die Installation erfolgreich war.

X:

Schließt das letzte Fenster.

Hiermit ist die Installation von Ghostscript beendet.

15.25. Die Klasse CCreatePDF

Diese Klasse erstellt Berichtsausgaben im PDF-Format. Als Parameter werden der Aliasname des zu verwendenden Cursors, der Name der zu erstellenden PDF-Datei, der Name der Berichtsdatei sowie eine optionale For-Klausel übergeben.

Um eine PDF-Datei erstellen zu können, müssen Ghostscript und ein Postscript-Druckertreiber auf dem jeweiligen Computer installiert sein. Diese Klasse prüft, ob Ghostscript bereits installiert ist. Sollte dies nicht der Fall sein, wird Ghostscript automatisch aus dem Internet heruntergeladen und installiert. Für das Herunterladen aus dem Internet wird die Klasse *CDownload* verwendet. In dem Memofeld *Install_gs* aus der Tabelle *Vfxsys.dbf* befindet sich das Skript, das zum Herunterladen und zur Installation von Ghostscript verwendet wird. In der Beschreibung der Klasse *CDownload* befinden sich weitere Hinweise.

Wenn kein Postscript-Druckertreiber installiert ist, installiert diese Klasse automatisch den Druckertreiber, dessen Name in der Eigenschaft *goProgram.PSPrinterToInstall* hinterlegt ist. In der Regel sind hierfür keine Benutzereingaben erforderlich.

Der Bericht wird über den Postscript-Druckertreiber ausgegeben und in einer Datei gespeichert. Das Programm Ghostscript wandelt diese Postscript-Datei in eine PDF-Datei um.

Eigenschaften

LastErrorNo – Diese Eigenschaft enthält die Nummer des letzten Fehlers (falls ein Fehler aufgetreten ist). Damit kann die Ursache des letzten Fehlers ermittelt werden.

LastErrorTest – Wenn ein Fehler aufgetreten ist, ist in dieser Eigenschaft der Text der Fehlermeldung zu finden.

Methoden

Create_PDF(tcAlias, tcRezFile, tcFRXName, tcFor)

tcAlias – Aliasname, der für die Berichtsausgabe verwendet wird.

tcRezFile – Vollständiger Pfadname der zu erstellenden PDF-Datei.

tcFRXName – Name der Berichtsdatei, die zur Erstellung der PDF-Datei verwendet wird.

tcFor – For-Klausel zur Filterung der zu exportierenden Daten.

Diese Methode gibt den Wert .T. zurück, wenn die PDF-Datei erfolgreich erstellt werden konnte. .F. wird zurückgegeben, wenn die PDF-Datei nicht erstellt werden konnte. In diesem Fall sind die Nummer und die Beschreibung des aufgetretenen Fehlers in den Eigenschaften *LastErrorNo* und *LastErrorText* gespeichert.

15.26. Die Klasse CEmail

Diese Klasse gibt dem Entwickler die Möglichkeit E-Mails zu versenden. Es müssen nur wenige Parameter der Methode *Send_Email_Report* übergeben werden um eine Berichtsausgabe im PDF-Format als E-Mail-Anhang versenden zu können.

Eigenschaften

LastErrorNo – Diese Eigenschaft enthält die Nummer des letzten Fehlers (falls ein Fehler aufgetreten ist). Damit kann die Ursache des letzten Fehlers ermittelt werden.

LastErrorTest – Wenn ein Fehler aufgetreten ist, ist in dieser Eigenschaft der Text der Fehlermeldung zu finden.

oEmail_Attachment – Diese Eigenschaft wird nur intern verwendet. Sie enthält eine Collection der Anhänge.

Methoden

AddAttachment (tsAlias, tcFileName, tcReport, tcFor)

Fügt dem E-Mail-Objekt Informationen über einen E-Mail-Anhang hinzu, der mit der nächsten E-Mail gesendet wird. Die Informationen über alle vorzubereitenden PDF-Anhänge werden in der Eigenschaft *oEmail_Attachment* gespeichert. Wenn der Aliasname einer geöffneten Tabelle oder Ansicht angegeben und der Name einer Berichtsdatei übergeben wird, wird diese Klasse automatisch eine PDF-Datei zu dem Bericht erstellen. Es kann ein weiterer Ausdruck als Parameter angegeben werden, der dazu verwendet wird die Daten des Berichts zu filtern. Wenn kein Aliasname angegeben wird und keine Tabelle im aktuellen Arbeitsbereich geöffnet ist, nimmt die Klasse an, dass ein Dateianhang vorbereitet wurde. In diesem Fall muss die Datei existieren, wenn die Methode *Send_Email_Report* aufgerufen wird.

tcAlias – Aliasname, der für die Berichtsausgabe und für den PDF-Export verwendet wird.

tcRezFile **–** Name des Dateianhangs (wenn eine PDF-Datei erstellt wird, wird dies der Name der PDF-Datei).

tcFRXName **–** Name der Berichtsdatei, aus der die PDF-Datei erstellt wird.

tcFor **–** For-Klausel mit der die Berichtsdaten für die PDF-Ausgabe gefiltert werden.

Send_Email_Report (tcEmail, tcSubject, tcText)
Sendet eine E-Mail. Wenn die E-Mail mit Anhängen versendet werden soll, müssen diese vorher mit der Methode *AddAttachment* angefügt werden.

tcEmail – Adresse des E-Mail-Empfängers.

tcSubject – Betreff der E-Mail.

tcText – Text der E-Mail.

ClearAttachment
Löscht alle E-Mail-Anhänge.

Die Methode *AddAttachment* kann entsprechend der Anzahl der benötigten Anhänge beliebig oft aufgerufen werden. Es werden die Aliasnamen der Tabellen oder Ansichten, die Namen der zu erstellenden Dateien, die Namen der Berichtsdateien und eventuell zu verwendende For-Klauseln als Parameter übergeben. Dann wird die Methode *Send_Email_Reports* aufgerufen. Alle PDF-Dateien werden erstellt und als E-Mail-Anhänge versendet. Auch die Dateien, die zuvor vorbereitet wurden und als Anhang versendet werden sollen, werden an die E-Mail angehängt.

15.27. Die Klasse CArchive

Diese Klasse dient der Datensicherung und Datenwiederherstellung. Die Daten werden in Zip-Archiven gesichert. Der Name des Archivs wird aus dem Namen des Datenordners und dem aktuellen Datum in ANSI-Form zusammengesetzt. Wenn zum Beispiel der Datenordner „Data“ heißt und die Datensicherung am 4. November 2004 durchgeführt wird, heißt das Archiv Data20041104.zip.

Eigenschaften

OverrideFile – Mit dieser Eigenschaft wird festgelegt was passiert, wenn eine Datei mit dem gleichen Namen schon vorhanden ist.

0 – Vorgang abbrechen, wenn bereits eine Datei mit dem gleichen Namen existiert.

1 – Wenn eine Datensicherung durchgeführt wird, werden neue Dateien dem Archiv hinzugefügt und bestehende Dateien werden aktualisiert. Wenn eine Wiederherstellung durchgeführt wird, werden existierende Dateien nicht überschrieben.

2 – Wenn eine Datensicherung durchgeführt wird, wird ein bestehendes Archiv überschrieben. Wenn eine Wiederherstellung durchgeführt wird, werden existierende Dateien überschrieben.

OperationSuccessfully – Enthält das Ergebnis der letzten Aktion.
.T. – wenn die Aktion erfolgreich ausgeführt werden konnte.

.F. – wenn die Aktion nicht ausgeführt werden konnte.

Methoden

CreateArchive (lcFileLocation, lcMask, lcArchFilePathName)
lcFileLocation – Vollständiger Pfad zu dem Ordner, dessen Inhalt gesichert werden soll.

lcMask – Zu sichernde Dateien, Beispiel: „*.DBF;*.FPT;*.CDX“.

lcArchFilePathName – Vollständiger Pfadname der zu erstellenden Archivdatei.

Rückgabewert: .T. – wenn die Aktion erfolgreich ausgeführt werden konnte, .F. – wenn die Aktion nicht ausgeführt werden konnte.

ZipProgress (*tcCurrentOperatedFile, nState, nAllFilesSize, nZIPedFilesSize, nArchiveCurrentSize)* Callback-Funktion der *CreateZipArchive*-Funktion (in *VFX.fll*).

tcCurrentOperatedFile – Der Name der Datei, die dem Archiv hinzugefügt wird.

nState – Aktuelle Aktion:

1 – Datei existiert
2 – Datei wird dem Archiv hinzugefügt
3 – Datei erfolgreich dem Archiv hinzugefügt
4 – Datei konnte dem Archiv nicht hinzugefügt werden
5 – Archivierungsvorgang erfolgreich beendet
6 – Archivierungsvorgang nicht erfolgreich beendet
7 – Keine Dateien zu archivieren

nAllFilesSize – Die Größe aller zu archivierenden Dateien.

nZIPedFilesSize – Die Größe der dem Archiv bereits hinzugefügten Dateien.

nArchiveCurrentSize – Die aktuelle Größe des Archivs.

Rückgabewert: 0 – Abbruch der Aktion. 1 – Fortsetzen Dateien dem Archiv hinzuzufügen und existierende Dateien zu überschreiben. 2 – Bestehende Archivdatei überschreiben

ExtractFromArchive(lcArchFileForExtract, lcPathForExtract)

lcArchFileForExtract – Vollständiger Pfadname der zu entpackenden Zip-Datei.

lcPathForExtract – Zielordner, in den die Dateien entpackt werden sollen.

UnZipProgress (tcCurrentOperatedFile, nState, nArchiveFilesSize, nUnZIPedFilesSize) Callback-Funktion der ExtractZipArchive-Funktion (in *VFX.fll*).

tcCurrentOperatedFile – Name der aktuell entpackten Datei aus dem Archiv.

nState – Aktuelle Aktion
1 – Datei existiert bereits
2 – Datei wird entpackt
3 – Datei entpacken beendet
4 – Datei konnte nicht entpackt werden
5 – Entpacken des Archiv erfolgreich abgeschlossen
6 – Entpacken des Archiv nicht erfolgreich abgeschlossen

nArchiveFilesSize – Größe des Archivs

nUnZIPedFilesSize – Größe des Teils des Archivs, das bereits entpackt wurde.

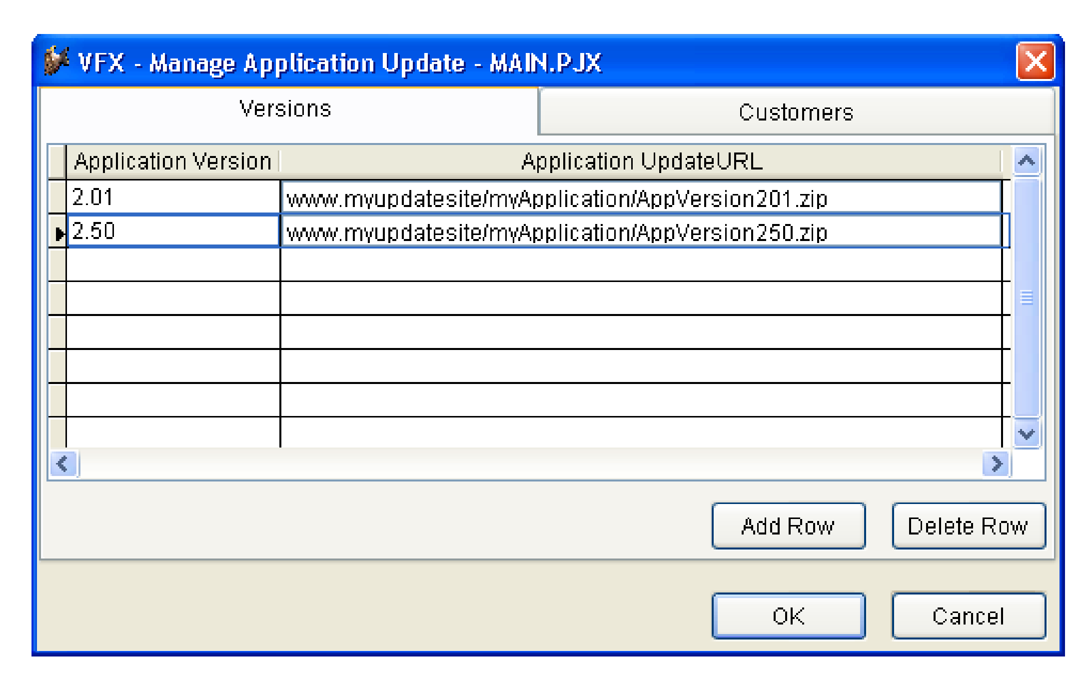
Rückgabewert: 0 – Abbruch der Aktion. 1 – Aktuelle Datei nicht entpacken.2 – Vorhandene Datei überschreiben.

15.28. Aktualisierung der Anwendung

Die Möglichkeiten zur Aktualisierung der Anwendung beim Kunden über das Internet wurden erweitert. Der Entwickler kann eine Liste der Kunden anlegen, die berechtigt ist, aktualisierte Programmversionen herunter zu laden und zu installieren. Diese Kundenliste wird in einer verschlüsselten Datei auf dem Web-Server gespeichert und vor der eigentlichen Aktualisierung auf den Kunden-PC heruntergeladen und geprüft. Zusammen mit der

Kundenliste wird eine Versionsliste heruntergeladen. Mithilfe dieser Versionsliste können abhängig von der beim Kunden installierten Programmversion unterschiedliche Aktualisierungen durchgeführt werden.

Beide Listen können aus dem VFX 11.0-Menü über den Menüpunkt *Activation, Manage Application Updates* bearbeitet werden.



In der Spalte *Application Version* wird die Nummer einer Anwendungsversion eingetragen. In der Spalte *Application Update URL* befindet sich der dazugehörige Download-Link.

Die Durchführung der Aktualisierung geschieht beim Kunden in zwei Schritten. Im ersten Schritt wird ein Download-Skript ausgeführt, das die Kundenliste und die Versionsliste herunterläd. Der Name der Datei mit der Kundenliste ist standardmäßig *UpdateCustomer.vfx*. Die Versionsliste heißt standardmäßig *UpdateVersion.vfx*. Das Download-Skript für diese beiden Dateien befindet sich in der Tabelle *Vfxsys.dbf* im Feld *UpdateApp*.

Nachdem die beiden Dateien heruntergeladen und entschlüsselt wurden, wird im zweiten Schritt geprüft, ob der Benutzer zur Aktualisierung berechtigt ist. Der Download-Link der für seine Anwendung geeigneten aktualisierten Version befindet sich in der Datei *UpdateVersion.vfx*.

15.29. VFP Toolbox für Entwickler

VFX unterstützt die Verwendung der VFP Toolbox für Entwickler. Wenn ein Projekt geöffnet wird, können die zu diesem Projekt gehörenden Klassen in die Toolbox geladen werden.

15.30. Die Weiterentwicklung mit VFP

Das gesamte VFX 11.0-Projekt liegt in normalen VFP Quelldateien vor. Die erstellte Anwendung kann also jederzeit mit VFP weiterentwickelt werden, auch wenn auf dem Entwicklungsrechner VFX nicht installiert ist.

15.31. Hilfe bei der Fehlersuche

Fehler „cap_application_title not found“: Eine Include-Datei wurde nicht gefunden. Stellen Sie sicher, dass der aktuelle Ordner der Ordner Ihres Projektes ist! **Tipp:** Geben Sie folgenden Befehl im Befehlsfenster ein: *CD ?*. Beenden Sie VFP, starten Sie VFP erneut, setzen Sie den aktuellen Pfad auf Ihren Projektordner, öffnen Sie Ihr Projekt, wählen Sie “Alle Dateien nochmals kompilieren” und starten Sie anschließend Ihr Projekt.

Hinweis: Wählen Sie die Option „Eigenschaften“ (letzte Option im Kontextmenü bei der Bearbeitung einer PRG-Datei) und wählen Sie „Vor dem Speichern kompilieren“. Dadurch haben Sie immer kompilierte PRG-Dateien.

Änderungen in den Include-Dateien werden nicht übernommen: Machen Sie eine Änderung in der Datei, die die Include-Datei einschließt, beenden Sie Visual FoxPro, löschen Sie alle kompilierten *FXP*-Dateien, starten Sie VFP erneut, wechseln Sie in den Projektordner und erstellen Sie das Projekt erneut. Tipp: Versuchen Sie auch den CLEAR PROGRAM-Befehl einzusetzen, der alle kompilierten Programme aus dem Speicher entfernt. Wenn Sie eine Änderung in einer Include-Datei machen, die von einem Formular eingeschlossen wird, öffnen Sie das Formular und speichern Sie es, sonst werden die Änderungen in der Include-Datei von dem Formular nicht übernommen. Wenn die Änderungen in Ihrer Include-Datei immer noch nicht wirksam werden, löschen Sie alle FXP-Dateien Ihres Projektes und wählen Sie „Alle Dateien neu kompilieren".

Wichtig! Aktueller Ordner: Stellen Sie sicher, dass der aktuelle Ordner der Ordner mit dem Projekt ist, mit dem Sie arbeiten! Versuchen Sie: *CD ?*

ANMERKUNG: Bevorzugen Sie die VFX Task Pane um Ihre Projekte zu öffnen.

Erstellte Formulare basieren nicht auf Bibliotheken aus dem Ordner meiner Anwendung: Dies ist nur dann ein Problem, wenn Sie gleichzeitig an verschiedenen Projekten oder an verschiedenen Versionen eines Projektes arbeiten. Um fehlerhafte Verweise zu beseitigen, benennen Sie vorübergehend den Ordner Ihres Projektes um. Öffnen Sie alle Formulare und Klassen und wählen Sie, falls erforderlich, die richtige Klassenbibliothek für Ihre Anwendung und speichern Sie die Änderungen.

Inkrementelle Suche und andere VFX-Grid-Eigenschaften funktionieren nicht: Stellen Sie sicher, dass Sie den VFX – CGrid Builder, wie in diesem Handbuch beschrieben, verwenden.

Die Eigenschaft „inkrementelle Suche" steht nicht zur Verfügung: Sie müssen den Puffermodus auf 3 setzen, da sonst keine IDX-Dateien angelegt werden können. Möglicherweise steht der Puffermodus bei Ihnen auf 5.

1:n-Formular zeigt die Daten der Child-Tabelle nicht an, wenn ich den Datensatzzeiger der Haupttabelle bewege: Prüfen Sie, ob Sie die 1:n-Beziehung in der Datenumgebung des Formulars richtig eingestellt haben! Sie müssen nur per drag & drop eine Beziehung vom Primärschlüssel der Haupttabelle zum Fremdschlüssel der Child-Tabelle ziehen. Ändern Sie keine anderen Eigenschaften. **Tipp: Setzen Sie nicht die OneToMany-Eigenschaft** Ihrer 1:n-Beziehung in der Datenumgebung Ihres Formulars auf wahr. Das Setzen dieser Eigenschaft auf wahr entspricht der Ausführung des SET SKIP TO-Befehls. Dieses Verhalten ist an dieser Stelle NICHT erwünscht.

Die Auswahlliste funktioniert nicht mit numerischen Feldern: Setzen Sie die Eigenschaft *cReturnExpr* der *CPickField*-Klasse auf *TRANSFORM(Feld)* anstatt auf *Feld*. Alles weitere funktioniert genauso wie bei Zeichenfeldern.

Änderungen in *PRG*-Dateien wirken sich nicht aus: Führen Sie den Befehl CLEAR PROGRAM aus und versuchen Sie es erneut. Oder setzen Sie besser die Bearbeitungsoption auf „Vor dem Speichern kompilieren".

Fehler beim Neuerstellen eines Projektes: Wenn Sie Probleme beim Neuerstellen eines Projektes haben, wählen Sie die „Rebuild"-Option aus der VFX Task Pane wie oben beschrieben.

ANMERKUNG: Die Include-Dateien und die Menüdateien sollten Sie von Hand überprüfen! Erwarten Sie nicht eine deutsche Anwendungsversion, wenn die Include-Dateien englisch sind.

15.32. Weitere Verbesserungen für Entwickler

- Aufruf aller VFX Form Builder auch vom Pageframe ausgehend möglich.
- Unterstützung von Ansichten und Cursoradapter bei der Anzeige des Audit Trails.
- Unterstützung von allen Steuerelementklassen in Buildern.
- Als Trennzeichen in allen VFX-Eigenschaften können jetzt wahlweise Komma oder Semikolon verwendet werden.
- Zusätzliche Felder *cins_time* und *cedt_time* zur Speicherung der letzten Bearbeitungszeit.
- Wenn *readonly=.T.* eingestellt ist, wird automatisch *tabstop=.F.* eingestellt.
- VFX – CPickfield Builder: die Eigenschaften *cfieldlist* und *cfieldtitle* sind auf dem Builder mit einer einfachen Textbox direkt erreichbar.
- VFX-Tabellen können wahlweise in einer SQL-Datenbank gespeichert werden.
- Neuer Builder zur Generierung von Audit-Trail-Triggern im DBC.

16. Fernwartung

In VFX 11.0 ist der Viewer-Teil des Fernwartungsprogramms Radmin integriert. Endanwender können die Fernwartung über den Menüpunkt Hilfe, Fernwartung starten. Die Fernwartung wird über das Internet durchgeführt.

16.1. Wie funktioniert die Fernwartung?

Zwischen dem Kunden-PC und dem Supporter-PC wird eine Verbindung über das IP-Protokoll aufgebaut. Standardmäßig wird der Port 4899 verwendet. VFX unterstützt ausschließlich IP-Verbindungen, die über das Internet hergestellt werden. Für IP-Verbindungen innerhalb eines LANs kann das Fernwartungsprogramm Radmin leicht manuell konfiguriert werden.

Um die Fernwartung nutzen zu können, muss der Kunden-PC über eine Internet-Verbindung verfügen. Die IP-Adresse muss über das Internet sichtbar sein. Der von Radmin verwendete Port 4899 darf nicht durch eine Firewall blockiert sein.

Zu den Vorteilen von Radmin gehört, dass keine Installation auf dem Kunden-PC notwendig ist. Für den Betrieb von Radmin sind auf dem Kunden-PC nur zwei Dateien erforderlich: *R_Server.exe* und *Admdll.dll*. Die Datei *R_Server.exe* kann aus einem beliebigen Ordner ausgeführt werden.

Bei der Einleitung der Fernwartung stellt der Kunden-PC eine Verbindung mit dem Internet her. In der Regel wird dem Kunden-PC beim Verbindungsaufbau mit dem Internet eine dynamische IP-Adresse zugewiesen. Dem Supporter kann diese IP-Adresse nicht bekannt sein. Die VFX-Anwendung beim Kunden registriert daher die aktuelle IP-Adresse des Kunden-PCs als Subdomain bei DynDNS. So kann der Supporter den Kunden-PC über einen Subdomain-Namen im Internet finden.

16.2. Voraussetzungen

Der Entwickler muss die VFX-Anwendung zunächst für die Fernwartung vorbereiten. Dafür muss zunächst eine Subdomain bei DynDNS für den Support der eigenen Anwendung angemeldet werden. Die Anmeldung ist kostenlos.

Die Registrierungsinformationen werden in der VFX-Anwendung in der Tabelle *Vfxsys.dbf* im Memofeld *dyndns* verschlüsselt gespeichert, damit die Registrierungsinformationen auf dem Kunden-PC nicht einsehbar sind. Die Verschlüsselung erfolgt mit dem Kennwort *cconfigpassword*. Dieses Kennwort muss in *Appl.vcx – CFoxAppl* in der Eigenschaft *cconfigpassword* eingetragen werden.

Die Bearbeitung der DynDNS-Registrierungsinformationen erfolgt über den Menüpunkt *Data, Manage Vfxsys.dbf* im *VFX 11.0*-Menü.

Der Inhalt des Memofeldes *dyndns* besteht aus vier Zeilen.

1. Benutzername bei DynDNS
2. Kennwort bei DynDNS
3. Subdomain-Name
4. Kennwort für den Radmin-Zugriff auf den Kunden-PC

16.3. Registrierung einer Subdomain

Über die Organisation Dynamic DNS Network Services ist es möglich kostenlos Subdomains zu registrieren. Jeder Entwickler sollte bei http://www.dyndns.org/services/dyndns eine dynamische DNS registrieren.

Für die Erstellung eines Kontos bei DynDNS sind ein Benutzername, ein Kennwort und eine E-Mailadresse erforderlich. Der Subdomain-Name kann beliebig gewählt werden. Es kann aus einer Vielzahl von Domain-Namen ausgewählt werden.

Beispiel: *meineFirma.dnsalias.com*

In diesem Beispiel ist *meineFirma* der selbst gewählte Subdomain-Name. *Dnsalias.com* ist der von DynDNS bereitgestellte Domain-Name.

Bei der Registrierung der Subdomain muss ein Benutzerkonto mit Benutzernamen und Kennwort angelegt werden. Mit den Anmeldedaten kann das Konto konfiguriert werden. Die Anmeldedaten sind auch in die obige URL einzusetzen.

Die dem Domain-Namen zugehörige IP-Adresse kann man beliebig oft und mit verschiedenen Methoden ändern. Ausführliche Beschreibungen zu allen Methoden finden sich auf der Website www.dyndns.org.

Die VFX-Anwendung ruft eine URL auf, um die aktuelle IP-Adresse des Kunden-PCs zu registrieren. Die URL hat das folgende Format:

```
http://benutzername:kennwort@members.dyndns.org/nic/update?hostname=mein
eFirma.dnsalias.com
```

Wenn man diese URL im Internet-Explorer eingibt, erhält man als Antwort eine HTML-Seite mit dem Wort „Good“.

Da der Internet-Browser die eigene IP-Adresse an den Server übermittelt, muss die IP-Adresse nicht gesondert angegeben werden. Der Internet-Server muss ja wissen an welche Adresse er die Antwort zurückschicken muss. Dyndns benutzt also automatisch diese IP-Adresse für die Registrierung der Subdomain.

16.4. Das Fernwartungsprogramm Radmin

Das Fernwartungsprogramm Radmin kann von der Website www.radmin.com herunter geladen werden. Auf dieser Website befindet sich auch die Dokumentation.

Radmin ist Shareware und kann kostengünstig registriert werden. Die Vollversion, die für den Supporter-Arbeitsplatz notwendig ist, kostet zurzeit 35 US$. Eine Lizenz für einen Kunden kostet 15 US$. Kundenlizenzen können nur in Paketen ab 50 Lizenzen erworben werden.

Ähnlich wie VFX ist auch Radmin über einen Aktivierungsschlüssel geschützt.

Wenn der Kunde die Fernwartung benutzen will, kann Radmin sofort verwendet werden. Wenn nach der 30-tägigen Testphase versucht wird eine Verbindung aufzubauen, wird der Supporter aufgefordert einen Registrierungsschlüssel zum Kundenrechner zu übertragen.

Der Registrierungsschlüssel kann während der Radmin-Verbindung vom Supporter an den Kundenrechner übertragen werden.

Neben der Fernwartung bietet Radmin die Möglichkeit zur Dateiübertragung.

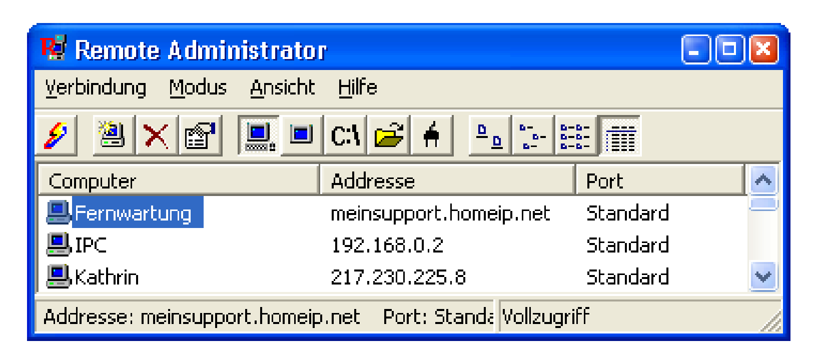
16.5. Die Fernwartung aus der Sicht des Supporters

Der Kunde sollte die Fernwartung nur nach Rücksprache mit dem Supporter starten. Das Fernwartungsprogramm ermöglicht den uneingeschränkten Zugriff auf den Kunden-PC und stellt für den Kunden damit ein erhebliches Sicherheitsrisiko dar!

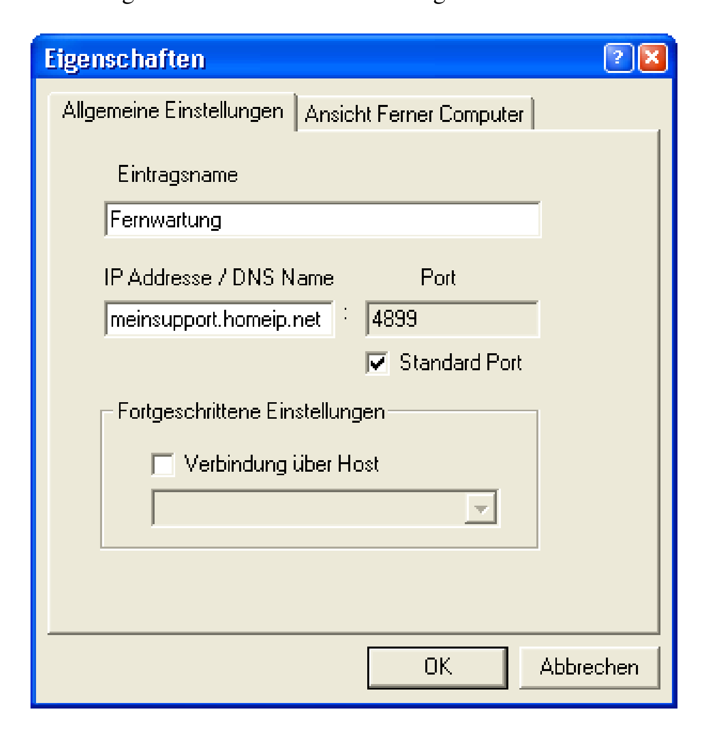
Der Zugriff auf den Kunden-PC sollte daher durch ein Kennwort geschützt werden.

Es ist nicht sehr wahrscheinlich, dass ein wartender Radmin-Server an einer dynamisch zugeteilten IP-Adresse im Internet von Hackern schnell gefunden wird. Zusätzlich ist der Zugriff auf den Kunden-PC durch ein Kennwort geschützt, das beim Verbindungsaufbau vom Supporter zum Kunden-PC eingegeben werden muss.

Im Remote Administrator Viewer wird ein Eintrag für den Support der Anwendung gemacht.



In den Eigenschaften des Remote-Eintrags wird im Feld IP-Adresse der Subdomain-Name eingetragen.



Der Kunden-PC kann jetzt über den Subdomain-Namen im Internet gefunden werden. Der Supporter braucht also nur einen einzigen Eintrag zur Fernwartung aller Kundenrechner.

Nach erfolgreicher Verbindungsherstellung kann der Kunden-PC im Fenster des Radmin-Viewers genau wie der eigene PC bedient werden.

17. Dokumentation

Neben dem Benutzerhandbuch gibt es zu VFX eine Menge an Online-Dokumentation. Dazu gehört insbesondere die Technische Referenz, die als Windows-Hilfedatei vorliegt. In ihr ist zu jeder Klassenbibliothek, zu jeder Klasse jede Methode und jede Eigenschaft beschrieben. In einem Tutorial werden anhand von typischen Anwenderfragen die Lösungen mit VFX erläutert. Direkt aus der Technischen Referenz können Videos (Avi-Dateien) gestartet werden. Es gibt 10 Videos mit insgesamt ca. 45 Minuten Dauer. In den Videos wird die Erstellung von Formularen für Fileserver- und Client-/Server-Datenbanken beschrieben und gezeigt. Für den VFX-Anfänger eine große Hilfe bei der Einarbeitung.

17.1. Support

Support für VFX ist im dFPUG-Forum (http://forum.dfpug.de) zu finden. Dort gibt es Sektionen zu VFX in deutscher, englischer und französischer Sprache. Diese Sektionen können auch alternativ als Newsgroup (news://news.dfpug.de) gelesen und bearbeitet werden.

Im Internet findet man auf der Website von Visual Extend (http://www.visualextend.de) weitere Informationen zum Produkt. Auch ist hier der Download der Demoanwendung, der gesamten Dokumentation und der aktuellen Vollversion von VFX möglich. Eine umfangreiche Sammlung weiterer Dokumente rund um VFX findet sich im Dokumentenportal der dFPUG (http://portal.dfpug.de). Aktuelle Informationen erhalten Sie über den kostenlosen dFPUG-eNewsletter im Abschnitt zu VFX (http://newsletter.dfpug.de).

18. Zusammenfassung

Wie wir gesehen haben stellt VFX eine vollständige Entwicklungsumgebung bereit, die keine Wünsche offen lässt. Alle wesentlichen Einstellungen an VFX-Klassen, insbesondere an den Formularklassen, können mit reentranten Buildern durchgeführt werden. Alle in diesem Artikel beschriebenen Eigenschaften und Funktionen lassen sich praktisch ohne Programmierung nur durch den Einsatz der Builder erreichen.

Trotzdem ist es an praktisch jeder Stelle über Hooks möglich in den Programmablauf einzugreifen.

Da VFX mit Quellen geliefert wird und selbst mit VFP programmiert ist, hat der Entwickler unbegrenzte Freiheit eigene Erweiterungen oder Anpassungen an eigene Bedürfnisse vorzunehmen.

Die Performance von VFX-Anwendungen ist so gut, wie sie mit VFP-Anwendungen nur sein kann. Die Vererbungstiefe ist gering. Die meisten Klassen haben nur 1 bis 2, maximal jedoch 5 Vererbungsebenen hinter sich. Um das Laden von umfangreichen Formularen weiter zu beschleunigen kann Delayed Instantiation verwendet werden. Auch dies wird von VFX mit einfach zu handhabenden Funktionen unterstützt.

Die mit VFX erstellten Anwendungen vermitteln dem Anwender einen sehr professionellen Eindruck und eine Office-kompatible Bedienung.

VFX bietet mit all dem ein unschlagbares Preis-/Leistungsverhältnis. Es bietet jedem Programmierer eine Fundgrube an Ideen und eine Vielzahl von fertigen Problemlösungen.

18.1. Ihre Meinung ist uns wichtig!

Senden Sie uns Ihre Meinung via eMail an visualextend@dfpug.de oder besuchen Sie unsere VFX Newsgroup unter news://news.dfpug.de.

Wir danken allen VFX-Kunden für das bisherige, großartige Feedback!

VFX 11.0 - Produktiver als je zuvor!

19. Neuheiten 2006 Q1 Neue Eigenschaften für Entwickler

19.1. Vererbungsarchitektur

19.1.1. Vfxobjbase.vcx

Die Vererbungsarchitektur der VFX-Klassen wurde erweitert. In bisherigen VFX-Versionen konnte nur mithilfe von Hooks in die Funktion und in das Layout von VFX-Basisklassen eingegriffen werden. Wenn ein Entwickler zum Beispiel in seiner gesamten Anwendung eine bestimmte Schriftart verwenden wollte, konnte dies nur über Hooks im *Init* Ereignis erreicht werden. Dies hatte den Nachteil, dass die Anwendung in den VFP-Designern stets in der VFX-Standardschriftart Arial angezeigt wurde und nur zur Laufzeit die über einen Hook eingestellte Schriftart angezeigt wurde.

In VFX 11.0 ist nun eine zusätzliche Vererbungsschicht vorhanden. Die in bisherigen VFX-Versionen vorhandenen VFX-Basisklassen aus der Klassenbibliothek *Vfxobj.vcx* befinden sich nun in der Klassenbibliothek *Vfxobjbase.vcx*. An die bisherigen Namen der Klassen wurde *base* als Suffix angehängt.

Zu jeder Klasse aus dieser Klassenbibliothek gibt es eine 1:1-Ableitung in der Klassenbibliothek *Vfxobj.vcx*. Die Klassenbibliothek *Vfxobj.vcx* steht nunmehr dem Entwickler für eigene Anpassungen und Erweiterungen zur Verfügung. Hier ist es beispielsweise möglich die Schriftart einer Klasse zu ändern. Diese Einstellung wirkt sich dann auf alle Steuerelemente basierend auf dieser Klasse in der gesamten Anwendung aus.

Bei einer Aktualisierung des Projekts mit dem VFX – Update Project Wizard wird die Klassenbibliothek *Vfxobj.vcx* nicht aktualisiert.

VFX-Formularklassen

Die VFX-Formulare verwenden jetzt keine Datenumgebung mehr. Das Design der VFX-Formularklassen wurde so erneuert, dass alle benötigten Datenquellen programmatisch geöffnet werden. Alle VFX-Formulare sind 1:1-Ableitungen aus den entsprechenden Formularklassen. Künftige Änderungen in VFX erfolgen also nur noch in den Formularklassen. Eine Änderung der VFX-Formulare ist nicht mehr erforderlich. Dadurch hat der Entwickler die Möglichkeit die VFX-Formulare nach eigenem Bedarf anzupassen, ohne dass bei einer Aktualisierung von VFX Änderungen verloren gehen bzw. noch mal gemacht werden müssen.

19.2. Datenzugriff

VFX-Anwendungen sowie alle Entwicklerwerkzeuge unterstützen SQL Server 2000 und die MSDE sowie SQL Server 2005 und SQL Server 2005 Express.

19.2.1. Einstellungen für die Datenumgebung

Viele SET-Einstellungen werden schon vor dem Öffnen von Cursorn in der Datenumgebung benötigt. Zusätzlich zu der Methode *SetDataEnvironment()* des Objekts *goEnvironment* können jetzt SET-Einstellungen für CursorAdapter-Objekte gemacht werden. Standardmäßig gelten jetzt für alle CursorAdapter-Objekte die gleichen SET-Einstellungen, die auch für Formulare gelten.

Die neue Methode *OnSetEnv* aus der Cursoradapter-Klasse *cBaseDataAccess* überprüft, ob das Formular, das den CursorAdapter instanziiert, eine Methode *OnSetEnv* besitzt. Wenn diese Methode existiert und wenn diese Methode nicht bereits von einem anderen CursorAdapter aufgerufen wurde, wird sie ausgeführt. Wenn das Formular keine *OnSetEnv*-Methode hat, wird überprüft, ob das Objekt *goEnvironment* existiert. Wenn dieses Objekt existiert, wird dessen Methode *SetDataEnvironment* aufgerufen. In allen anderen Fällen werden fest codierte SET-Einstellungen ausgeführt.

In der Klasse *cAppDataAccess* in der Klassenbibliothek *Appl.vcx* können eigene SET-Einstellungen hinzugefügt werden.

19.2.2. Das neue Objekt goPath

Zur Laufzeit einer Anwendung wird das neue Objekt *goPath* instanziiert. Dieses Objekt hat Eigenschaften, deren Werte auf die aktuell verwendeten Pfade zeigen:

CDataDir – Pfad zur aktuell verwendeten Datenbank.

ClientName – Name der aktuellen Datenbank. Dies ist der für den Benutzer im Mandantenauswahldialog sichtbare Name. Dies ist nicht unbedingt der physikalische Name der Datenbank.

VfxPath – Pfad zu den VFX Systemtabellen.

ReportPath – Pfad zu den Berichtsdateien.

UpdatePath – Pfad zu dem Ordner mit einer aktualisierten Datenbankstruktur.

ImportPath – Diese Eigenschaft wird von VFX nicht verwendet und steht zur freien Verfügung.

ExportPath – Diese Eigenschaft wird von VFX nicht verwendet und steht zur freien Verfügung.

Wenn die Tabelle *Vfxpath.dbf* zur Mandantenauswahl verwendet wird, wird jedes Feld dem Objekt goPath als Eigenschaft hinzugefügt.

Wenn die Datei *Config.vfx* zur Mandantenauswahl verwendet wird, wird jedes Feld dem Objekt goPath als Eigenschaft hinzugefügt.

So kann zur Laufzeit auf einfachem Weg auf die aktuellen Pfadeinstellungen zugegriffen werden.

19.2.3. Erweiterungen in der Klasse cBaseDataAccess

Neue Eigenschaften

cFieldsToWriteNULLWhenEmpty – Dieser Eigenschaft kann eine Liste von Feldern zugewiesen werden, deren Werte mit *NULL* ersetzt werden, bevor Daten in die Datenbank geschrieben werden.

cForeignKeyName – Diese Eigenschaft wird nur bei Child-CursorAdaptern auf OneToMany-Formularen verwendet. Hier wird der Name des Feldes im Parent-Arbeitsbereich gespeichert, das den Primärschlüssel enthält.

cForeignKeyValue – Diese Eigenschaft wird nur bei Child-CursorAdaptern auf OneToMany-Formularen verwendet. Mit diesem Ausdruck wird der Wert des Fremdschlüssels des neu gespeicherten Parent-Datensatzes ermittelt. Normalerweise ist dies das Feld mit dem Primärschlüssel aus dem Parent-Arbeitsbreich. Das Ergebnis der Evaluierung dieses Ausdrucks wird in dem Feld gespeichert, das in der Eigenschaft *cForeignKeyName* angegeben ist.

cWhereClause – Hier kann eine Where-Klausel angegeben werden. Zur Laufzeit wird dieser Wert an den Wert aus der Eigenschaft *SelectCmd* angefügt. Anschließend wird die *CursorFill*-Methode aufgerufen. Da dieser Wert auf Formularebene eingegeben werden kann, bleibt die CursorAdapter-Klasse unverändert. Änderungen an der CursorAdapter-Klasse, insbesondere wegen Strukturänderungen der Basistabellen, können unabhängig von der Where-Klausel durchgeführt werden.

lWriteNULLWhenEmptyForDBC – Wenn der Wert dieser Eigenschaft mit *.T.* eingestellt wird, werden leere Felder der Typen *Date* und *Datetime* mit dem Zustand *NULL* in der Datenbank gespeichert. Hierdurch können leere Datumswerte in einer Datenbank gespeichert werden, die später in eine SQL Datenbank portiert werden soll. Der Standardwert dieser Eigenschaft ist *.F.* Diese Eigenschaft wird nur verwendet wenn der *DataSourceType* eines CursorAdapter *Native* ist.

Neue Methode

OnSetEnv- Diese Methode wird während der Initialisierung aufgerufen.

Die neue Methode *OnSetEnv* überprüft, ob das Formular, das den CursorAdapter instanziiert, eine Methode *OnSetEnv* besitzt. Wenn diese Methode existiert und wenn diese Methode nicht bereits von einem anderen CursorAdapter aufgerufen wurde, wird sie ausgeführt. Wenn das Formular keine *OnSetEnv*-Methode hat, wird überprüft, ob das Objekt goEnvironment existiert. Wenn dieses Objekt existiert, wird dessen Methode SetDataEnvironment aufgerufen. In allen anderen Fällen werden fest codierte SET-Einstellungen ausgeführt.

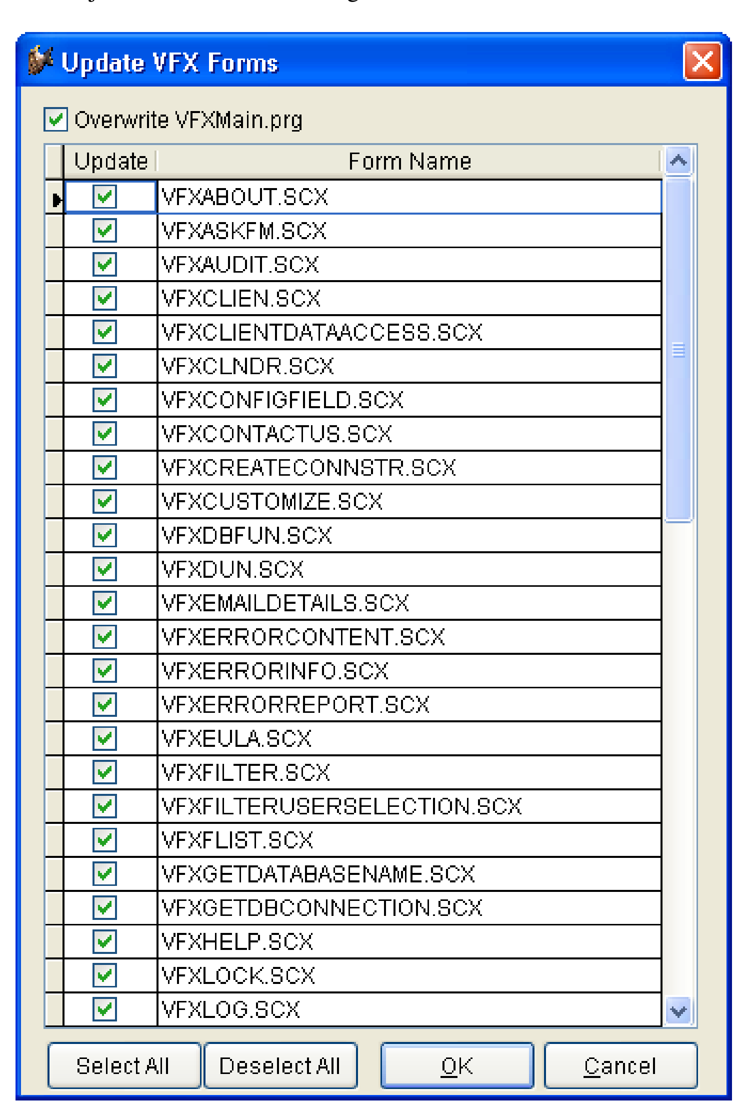
19.3. Builder und Wizards

19.3.1. VFX – Task Pane

In der VFX – Task Pane befindet sich in der Symbolleiste eine neue Schaltfläche. Hierüber kann der VFX – Update Project Wizard gestartet werden.

19.3.2. VFX – Update Project Wizard

Vor der Aktualisierung wird eine Archivdatei vom Projektordner mit allen Unterordnern angelegt. Der Dateiname besteht aus dem Projektnamen, dem aktuellen Datum im ANSI-Format sowie der VFX-Version, die das Projekt vor der Aktualisierung hat.



In einem anschließend erscheinenden Dialog kann eingestellt werden, welche der VFX-Formulare bei der Aktualisierung überschrieben werden sollen. Entwickler, die in VFX-Formularen Änderungen gemacht haben, dürfen diese Formulare nicht überschreiben. Die Einstellungen in diesem Dialog werden für spätere Aktualisierungen gespeichert und bleiben so erhalten.

19.3.3. VFX – Application Builder

Alle neuen Eigenschaften der Klassen *cFoxAppl* und *cAppUpdateEngine* können im VFX – Application Builder bearbeitet werden.

Zu jeder Eigenschaft wird im Builder ein Tooltip mit dem Namen der bearbeiteten Klasse und Eigenschaft im Format *Klasse.Eigenschaft* angezeigt.

Der VFX – Application Builder wurde um eine Suchfunktion erweitert. Damit ist es möglich nach jedem Text, der in einer Bezeichnung im Builder vorkommt, zu suchen.

19.3.4. VFX – Upsizing Wizard

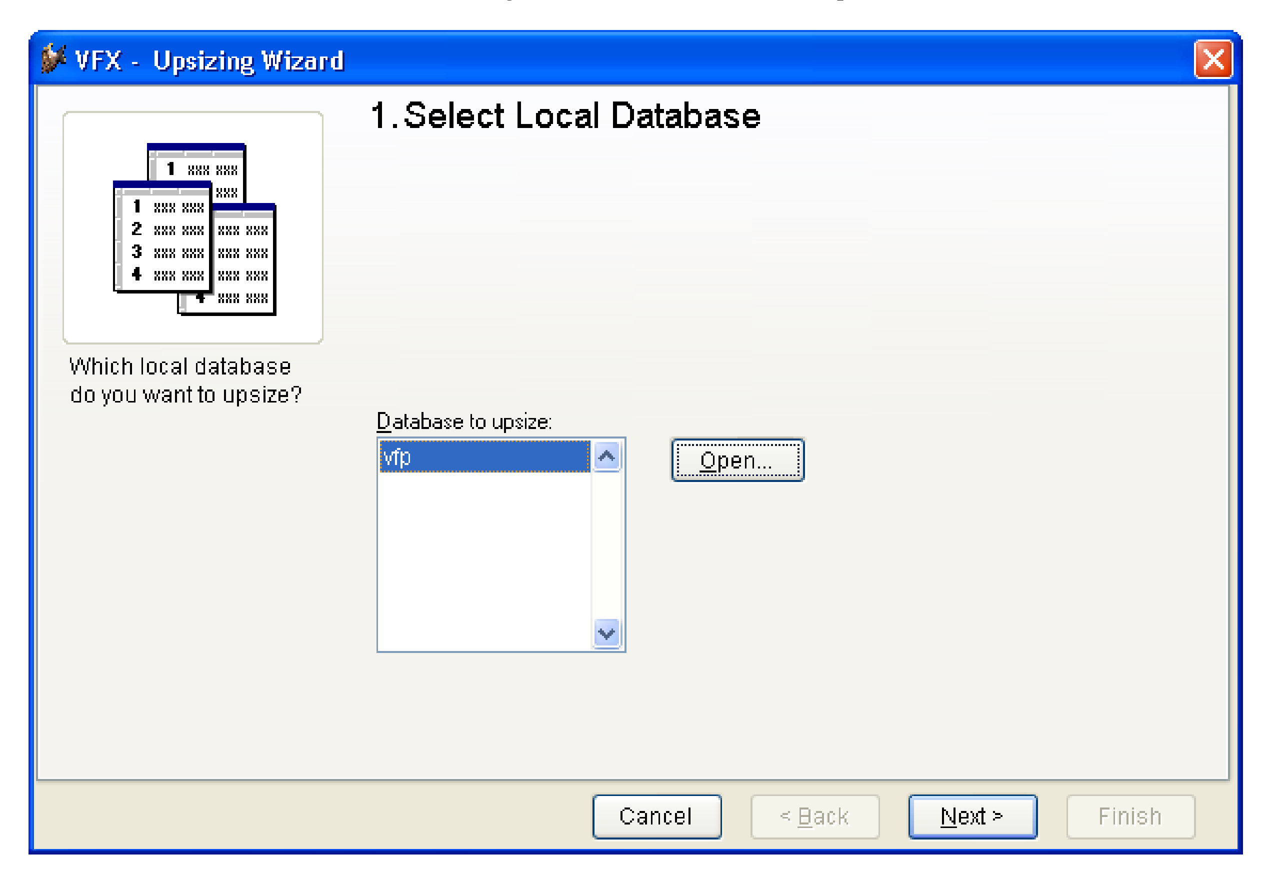
Der VFX – Upsizing Wizard entspricht in seinem Funktionsumfang etwa dem VFP – Upsizing Wizard. Mit dem VFX – Upsizing Wizard können jedoch auch ID-Werte korrekt in eine Remote Datenbank portiert werden. Die Portierung von Daten ist ohne Codepage-Konflikte möglich. Auch Feldnamen, die im SQL Server geschützte Befehlsworte sind, können portiert werden. Standardmäßig sind in allen Feldern Nullwerte erlaubt. Im Gegensatz zum VFP – Upsizing Wizard wird standardmäßig eine neue Datenbank angelegt und nicht vorgeschlagen eine vorhandene Datenbank zu überschreiben. Und schließlich erfordert der VFX – Upsizing Wizard keine DSN-Verbindung, sondern erlaubt beliebige Verbindungswege zur Remote Datenbank.

Der VFX – Upsizing Wizard ermöglicht es eine vorhandene VFP-Datenbank auf einen SQL Server zu portieren. Dabei wird die Struktur so genau wie möglich auf dem SQL Server abgebildet. Die Daten werden in die neue SQL Server-Datenbank übernommen. Auch die Portierung von Ansichten wird unterstützt.

Der VFX – Upsizing Wizard führt den Entwickler durch sechs Schritte:

1. Auswahl der lokalen Datenbank

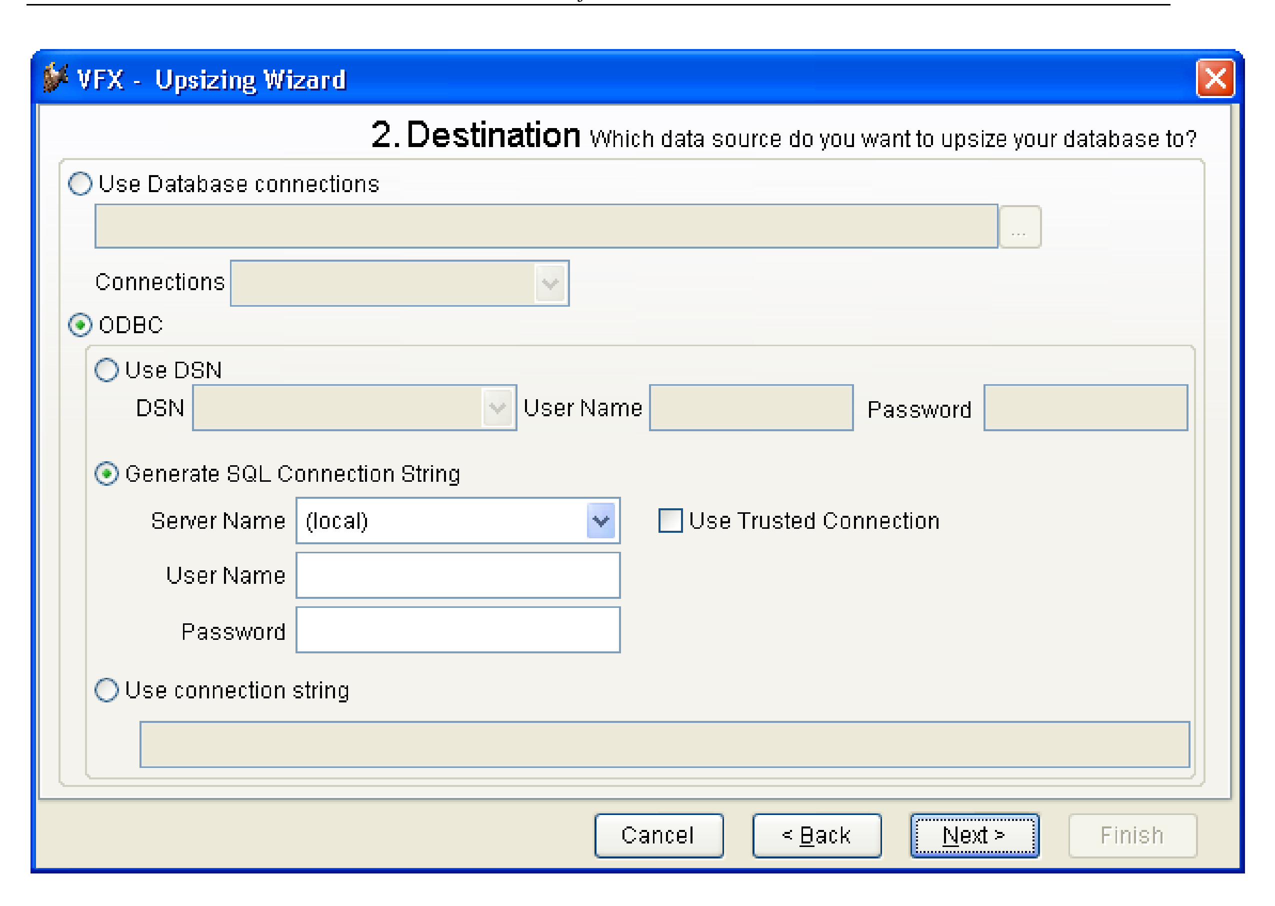
Im ersten Schritt wird die lokale Datenbank ausgewählt, die auf den SQL Server portiert werden soll.



Es wird eine Liste der zurzeit geöffneten Datenbanken angezeigt. Wenn die zu portierende Datenbank nicht geöffnet ist, kann sie hier über die Schaltfläche *Open* geöffnet werden.

2. Ziel

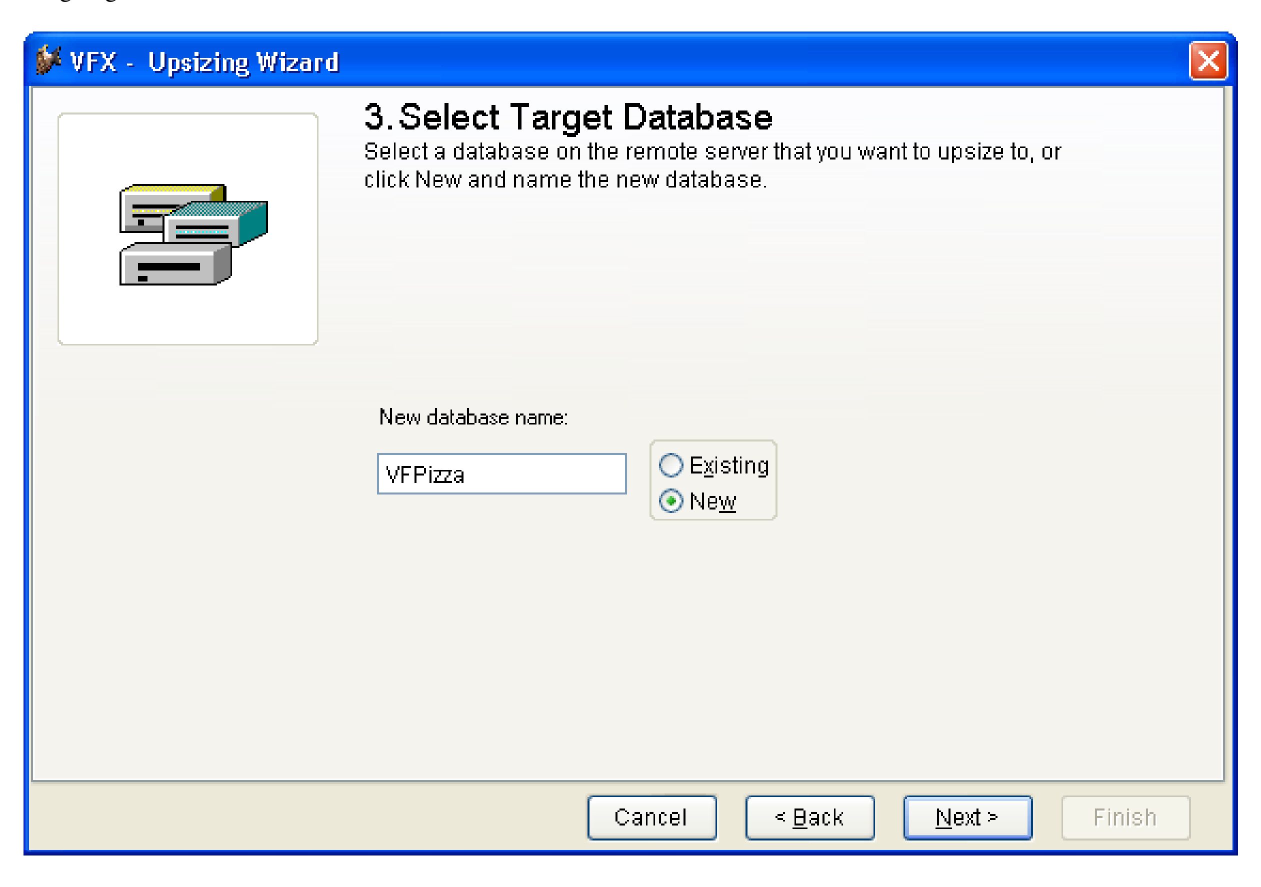
In diesem Schritt wird die Verbindung zum SQL Server angegeben. Es kann eine vorhandene Verbindung aus einem DBC verwendet werden. Es kann aber auch eine vorhandene DSN-Verbindung oder eine vorhandene Verbindungszeichenfolge verwendet werden.



Mit dem Wizard für Verbindungszeichenfolgen kann ebenfalls eine Verbindung hergestellt werden. Hierbei wird standardmäßig der lokal installierte SQL Server vorgeschlagen. Wenn der eingegebene Benutzername mit dem Kennwort nicht zu einer erfolgreichen Anmeldung führt, wird automatisch versucht eine vertrauenswürdig Verbindung mit den Windows-Anmeldedaten herzustellen. In der Regel sind auf dieser Seite des Assistenten also keine Eingaben erforderlich.

3. Eingabe des Datenbanknamens

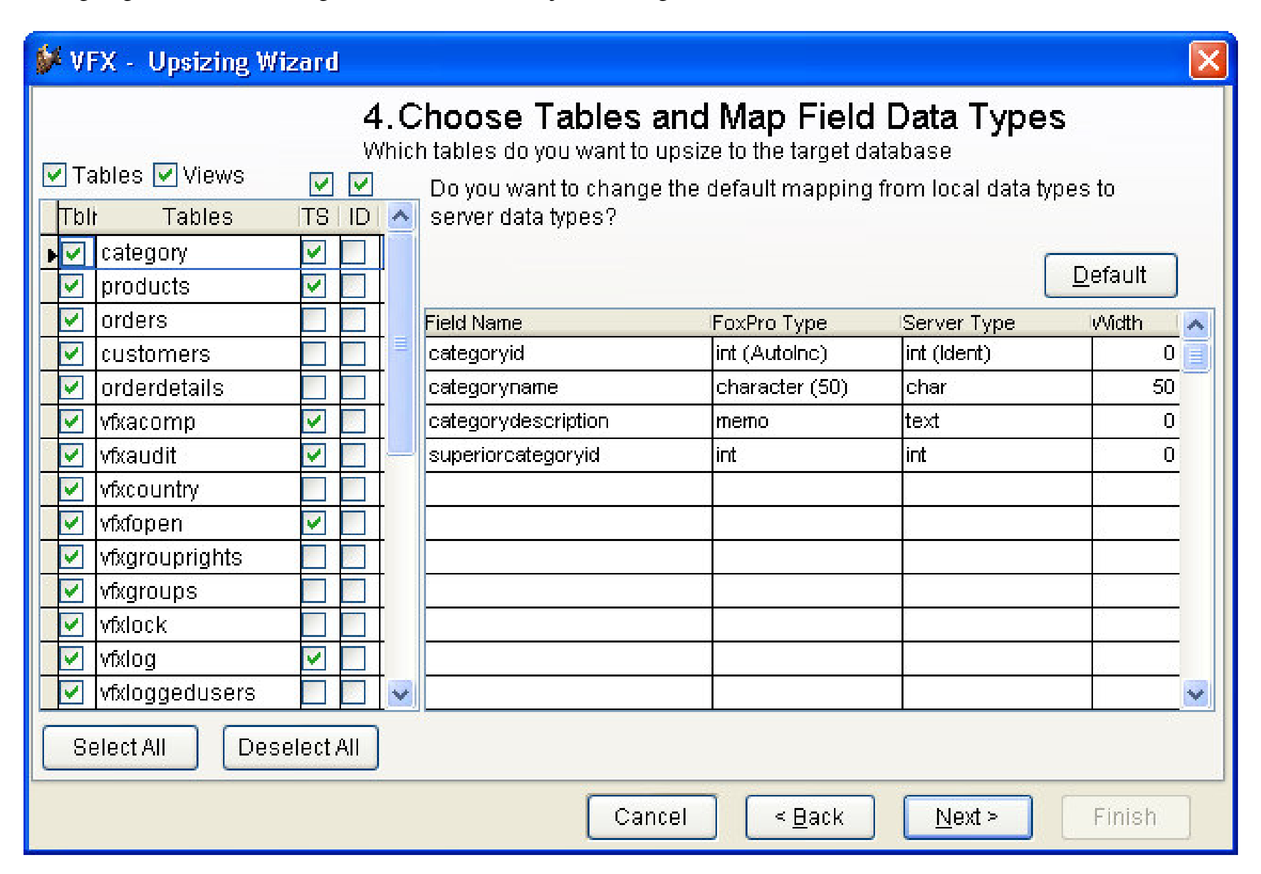
Wenn die Option *New* ausgewählt ist, kann der Name der neuen Datenbank eingegeben werden. Der Name muss ein gültiger Name für eine remote Datenbank sein.



Wenn die Option *Existing* ausgewählt wird, wird eine Liste der auf dem SQL Server vorhandenen Datenbanken angezeigt. Wählen Sie die Datenbank aus, die vom VFX – Upsizing Wizard aktualisiert werden soll.

4. Auswahl der Tabellen und Zuordnung der Datentypen

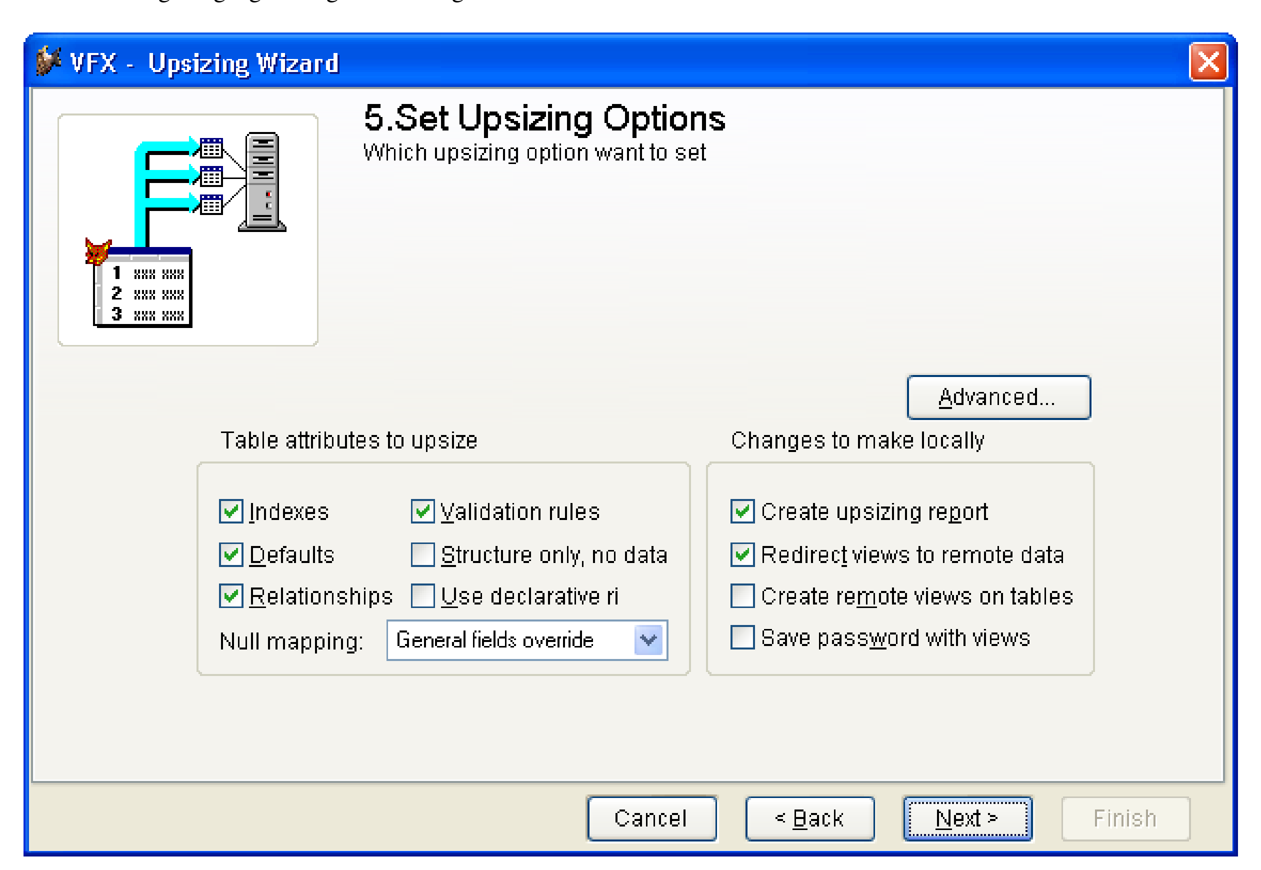
Hier kann ausgewählt werden welche Tabellen aus der VFP-Datenbank auf den SQL Server portiert werden sollen. Standardmäßig sind alle Tabellen zur Portierung markiert. Jeder Tabelle kann ein Timestamp-Feld sowie ein ID-Feld hinzugefügt werden. Der VFX – Upsizing Wizard fügt Tabellen mit Memo-Feldern standardmäßig ein Timestamp-Feld hinzu. Tabellen, die kein Primärschlüsselfeld enthalten, wird automatisch ein ID-Feld hinzugefügt. Diese Einstellungen können bei Bedarf je Tabelle geändert werden.



Zur selektierten Tabelle werden die Struktur in der VFP-Datenbank sowie die Struktur in der zu erstellenden SQL Server-Datenbank angezeigt. In der Regel wird für jeden Feldtyp eine sinnvolle Portierung vorgenommen. Bei Bedarf kann die Zuordnung des Datentyps hier geändert werden.

5. Portierungsoptionen

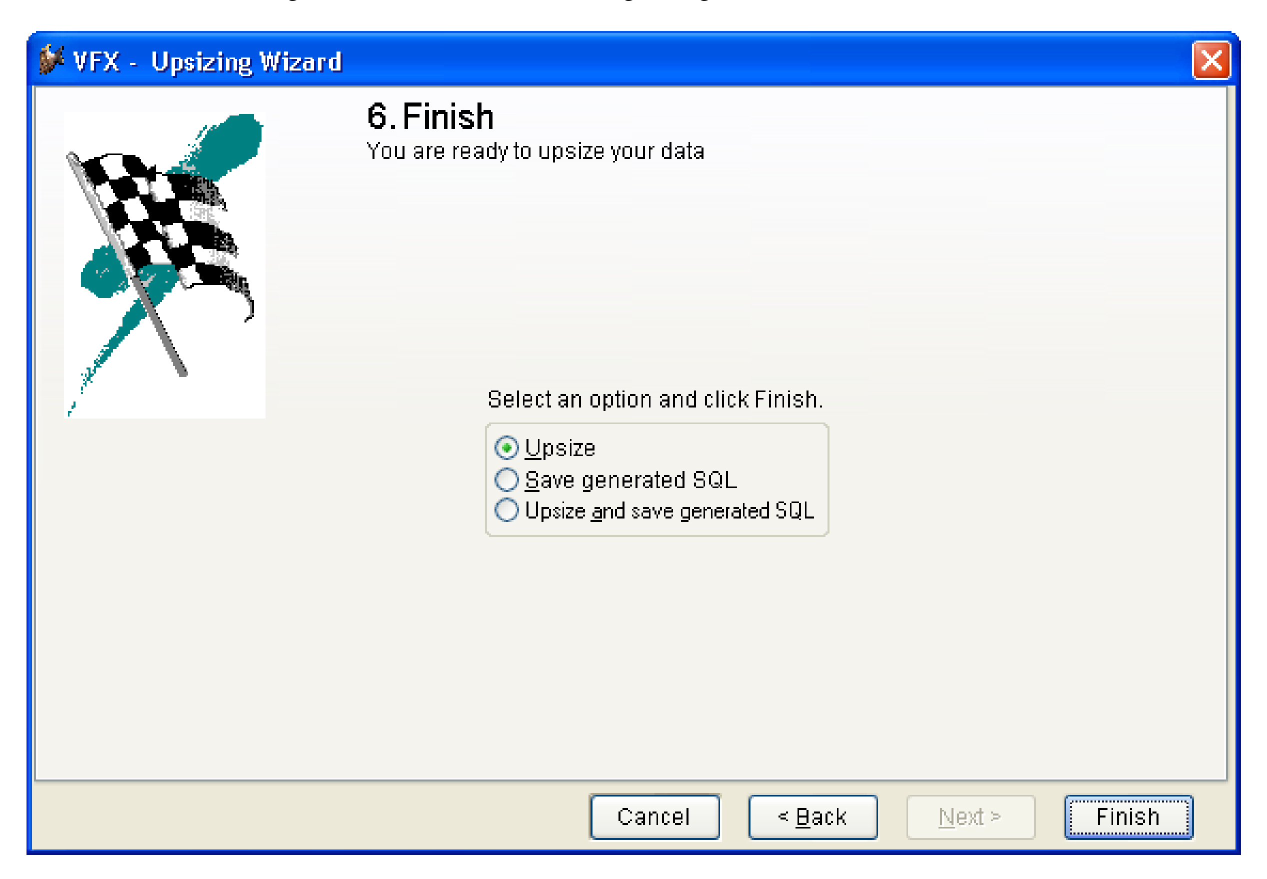
Standardmäßig werden die Strukturen von Tabellen sowie die Daten portiert. Es können auch Indexschlüssel, Standardwerte, Beziehungen (RI Constraints) und Validierungsregeln portiert werden. In der Combobox *Null mapping* kann eingestellt werden, ob Nullwerte erlaubt sind. Diese Option hilft sicherzustellen, dass Einfüge- und Aktualisierungsvorgänge erfolgreich durchgeführt werden können.



In diesem Schritt kann insbesondere auch eingestellt werden, ob ein Bericht über die Portierung erstellt werden soll. Der Bericht wird in ein neues VFP-Projekt eingefügt. Aus dem Bericht sind Probleme bei der Portierung ersichtlich.

6. Fertig

In diesem Schritt kann eingestellt werden, wie die Portierung durchgeführt werden soll.



Es kann eine der Optionen gewählt werden:

Upsize– Führt die Portierung wie oben beschrieben durch.

Save generated SQL– Generiert SQL Befehle, die für die Portierung erforderlich sind. Durch Ausführung dieser Befehle kann die eigentliche Portierung zu einem späteren Zeitpunkt durchgeführt werden.

Upsize and save generated SQL– Führt die Portierung wie oben beschrieben durch und generiert zusätzlich SQL Befehle, um die Portierung zu einem späteren Zeitpunkt wiederholen zu können.

Es ist eine gute Idee vor der Portierung eine Datensicherung durchzuführen. Während der Portierung werden Tabellen und lokale Ansichten aus der VFP-Datenbank umbenannt, um Tabellen und remote Ansichten mit den gleichen Namen in der SQL Server-Datenbank erstellen zu können.

Der VFX – Upsizing Wizard erlaubt es Felder vom Typ *Date* und *Datetime* mit leeren Werten in eine SQL Datenbank zu portieren. Wenn diese Felder in der SQL Datenbank den Zustand *NULL* erlauben, wird *NULL* in die SQL Datenbank geschrieben. Wenn der Zustand *NULL* nicht erlaubt ist, wird *01.01.1900* in die SQL Datenbank geschrieben.

19.3.5. VFX – CursorAdapter Wizard

CursorAdapter-Klassen können jetzt auch basierend auf Ansichten, gespeichert in einem DBC oder einer SQL Server Datenbank, erstellt werden.

Der Wizard kann jetzt auch auf einzelne Tabellen oder Ansichten angewendet werden.

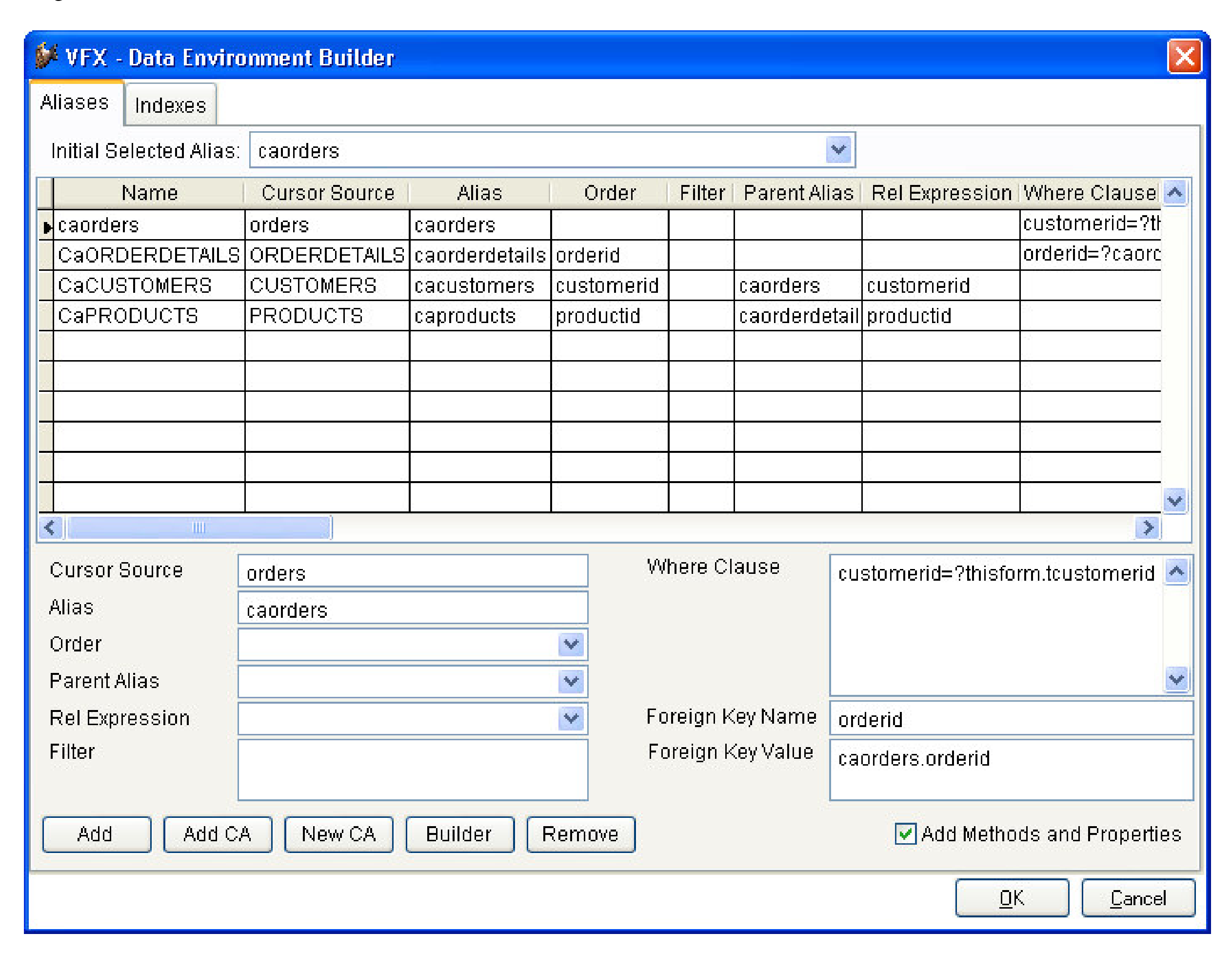
Für jedes Feld aus jeder Tabelle kann eine Typkonvertierung durchgeführt werden. Es stehen alle VFP-Feldtypen zur Verfügung.

Der VFX – ConnectionString Wizard steht jetzt auch im VFX – CursorAdapter Wizard zur Verfügung.

Der VFX – CursorAdapter Wizard fügt der Datei *Config.vfx* automatisch einen Verbindungseintrag zur ausgewählten Remote Datenbank hinzu.

19.3.6. VFX – Data Environment Builder

Der VFX – Data Environment Builder kann auch als eigenständiger Builder auf *Dataenvironment*-Klassen eingesetzt werden. Bereits in VFX 9.0 wurden *Dataenvironment*-Klassen für Formulare unterstützt.



Unterhalb des Grid können jetzt alle Einstellungen zum aktuellen Cursor zusätzlich in Textboxen, Editboxen und Comboboxen bearbeitet werden. Längere Zeichenketten, wie zum Beispiel für einen Filterausdruck oder eine Where-Klausel können so einfacher bearbeitet werden.

Im VFX – Data Environment Builder gibt es einige neue Einstellmöglichkeiten:

- *Where Clause* – Dieser Wert wird nur bei Verwendung von CursorAdaptern berücksichtigt. In dieser Spalte kann eine Where-Klausel eingetragen werden. Der eingegebene Wert wird in der Eigenschaft *cWhereClause* des CursorAdapters gespeichert. Zur Laufzeit wird diese Where-Klausel automatisch dem Select-Befehl aus der Eigenschaft *SelectCMD* eines CursorAdapters hinzugefügt, bevor der CursorAdapter mit Daten gefüllt wird. Dies hat den Vorteil, dass die Eigenschaft *SelectCMD* nicht auf Formularebene verändert werden braucht und der Wert in jedem Fall aus der basierenden CursorAdapter-Klasse vererbt wird. Bei einer Veränderung der Struktur der zugrunde liegenden

Tabellen kann die CursorAdapter-Klasse mit dem VFX – CursorAdapter Wizard aktualisiert werden. Änderungen an Formularen sind nicht mehr erforderlich.

Wenn mit CursorAdaptern gearbeitet wird und Primärschlüssel von der Datenbank vergeben werden, müssen in einem 1:n Szenario die vergebenen Primärschlüssel für den Parent-Datensatz als Fremdschlüssel in den Child-Datensätzen gespeichert werden. Dazu ist es erforderlich, dass zuerst der Parent-Datensatz gespeichert wird und nach dem Speichervorgang der Primärschlüssel aus der Datenbank gelesen wird. Im CursorAdapter für die Child-Daten muss der Fremdschlüssel spezifiziert werden. VFX speichert den Fremdschlüssel automatisch in allen Datensätzen eines Child-CursorAdapters. Zu diesem Zweck gab es bereits in VFX 9.0 in der VFX-CursorAdapter-Klasse die Eigenschaften *cForeignKeyName* und *cForeignKeyValue*. In VFX 11.0 werden die Werte dieser Eigenschaften automatisch ermittelt und vorbelegt und können im VFX – Data Environment Builder bearbeitet werden.

In VFP-Datenbanken kann zur Erzeugung von Primärschlüsseln der Datentyp *Integer (Autoinc)* und bei SQL Server-Datenbanken der Datentyp *Integer Identity* verwendet werden.

- *Foreign Key Name* – Hier wird der Name des Feldes angegeben, in dem der Fremdschlüssel gespeichert werden soll. Der Name des Feldes wird in der Eigenschaft *cForeignKeyName* gespeichert.

- *Foreign Key Value* – Hier wird der Name des Feldes aus dem Parent-CursorAdapter angegeben, das den neuen Primärschlüssel nach dem Speichern enthält. Der hier eingegebene Wert wird in der Eigenschaft *cForeignKeyValue* gespeichert. Hier kann auch ein Ausdruck eingegeben werden. Dieser Ausdruck wird evaluiert und dem in der Eigenschaft *cForeignKeyName* eingetragenen Feld zugewiesen.

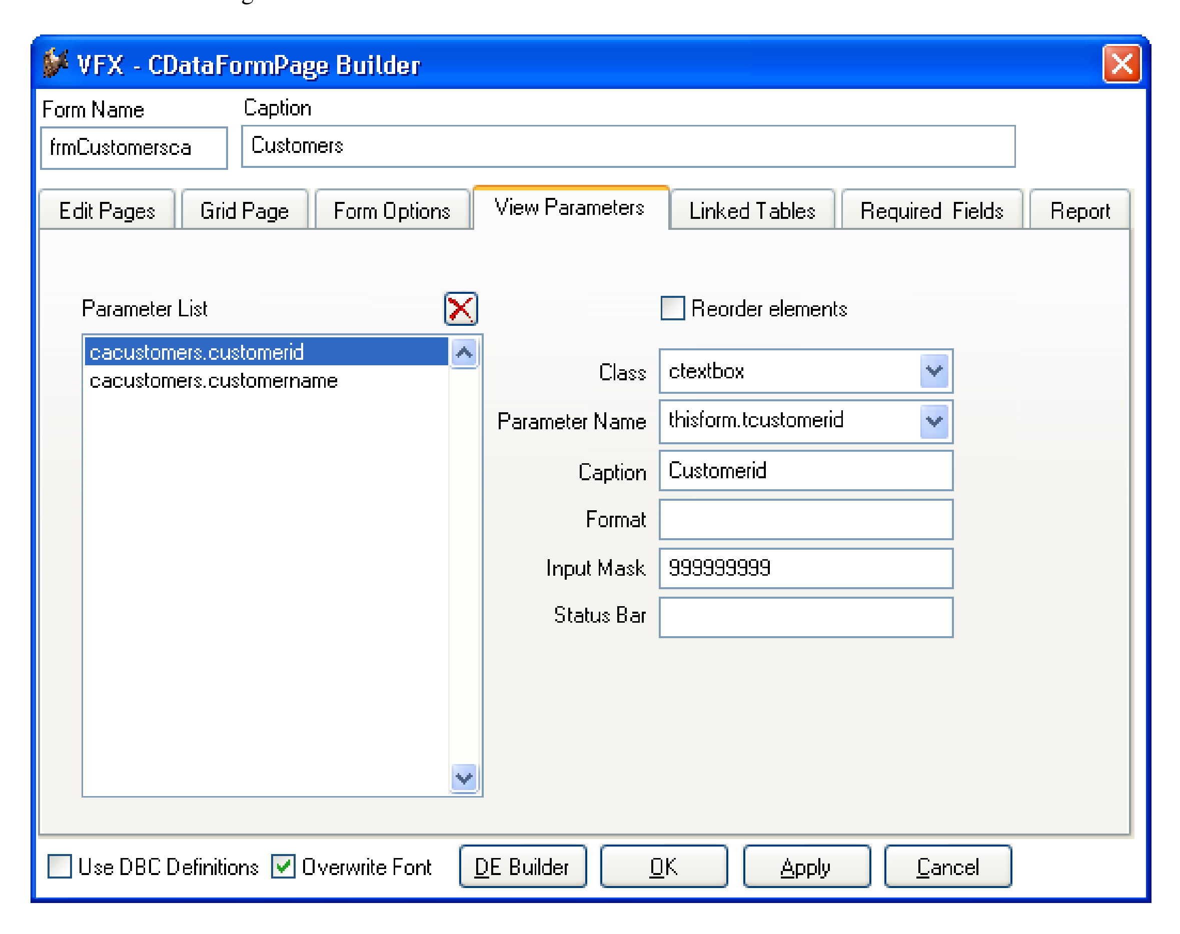
Wenn CursorAdapter verwendet werden, die nicht auf der VFX-Klasse *cAppDataAccess* basieren, aber die Eigenschaften des VFX – Data Environment Builder trotzdem genutzt werden sollen, kann das Kontrollkästchen *Add Methods and Properties* markiert werden. Hierdurch werden dem CursorAdapter die benötigten Eigenschaften automatisch hinzugefügt.

19.3.7. VFX – Form Builder

Wenn man im VFP – Formular-Designer den Builder startet, wird direkt der VFX – Form Builder gestartet. Der VFX – Data Environment Builder wahlweise durch einen Klick auf die Schaltfläche *DE Builder* oder direkt aus der Datenumgebung des VFP – Formular-Designers gestartet werden.

In allen Formular Buildern kann die basierende Klasse für Steuerelemente zur Eingabe von Ansichtsparametern ausgewählt werden.

Die in der Eigenschaft *cWhereClause* verwendeten Parameter eines CursorAdapters können aus der Combobox *Parameter Name* ausgewählt werden. Zusätzlich stehen in dieser Combobox



Die Einstellungen in den VFX – Form Buildern *Is Child Form* und *Has linked child form* werden nicht mehr benötigt und sind redundant. Jedes Formular kann ohne besondere Einstellung als Parent- und als Child-Formular dienen.

Wenn Formulare basierend auf CursorAdaptern als Datenquellen erstellt werden, lesen die VFX – Form Builder jetzt die Eigenschaften aus den zugrunde liegenden Tabellen aus, wenn die Datenbanken zur Verfügung stehen.

Die VFX – Form Builder erstellen jetzt bei Seitenrahmen lokalisierte Überschriften für alle Seiten.

Eine Vielzahl neuer Eigenschaften kann auf der Seite *Form Options* der Form Builder eingestellt werden.

VFX - COneToMany Builder

Form Name: frmOrdersca
Caption: AuftragCA
Master Table: caorders

Edit Pages | Grid Page | Form Options | Children | View Parameters | Linked Tables | Required | Report

Report Name ...
Arial,9,N

Auto Sync. Child Form
Put In Last File Menu
Put In Window Menu
Multi Instance
Close with ESC Key
Copy Child
Child Alias
cacustomers

Can Edit
Can Insert
Can Copy
Can Delete
Can Export
Hide When Empty
Auto Edit
Edit on Enter
Ask To Save
Allow Save Empty Records

Save/Restore Positions
Add SpeedBar Control
Enable Child Insert on Click
Search On Init
On Search Use Grid
Multiline Report
Use Custom Print Dialog
Use Report Behavior 80 for PDF

Search Form: VFXSRCH
Filter Behavior: 1 - VFX90
Start Page: -1

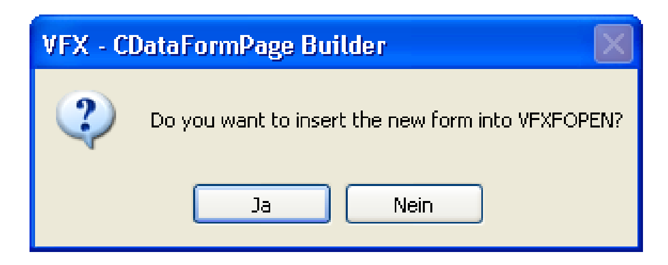
Favorites
Favorite Description
Key field
Caption of the menu
SCX file name

Use DBC Definitions
Overwrite Font
DE Builder | OK | Apply | Cancel

Mit dem Kontrollkästchen *Allow Save Empty Records* kann eingestellt werden, ob neue, leere Datensätze beim Speichern automatisch und ohne Rückfrage verworfen werden sollen. Entsprechend der Markierung wird der Wert der Formulareigenschaft *lAllowSaveEmptyRecords* gesetzt. Der Standardwert ist *.T.*, leere Datensätze dürfen gespeichert werden.

Alle Einstellungen, um einen Datensatz dem Favoriten-Menü hinzuzufügen, können jetzt in Textboxen auf der Seite *Form Options* gemacht werden.

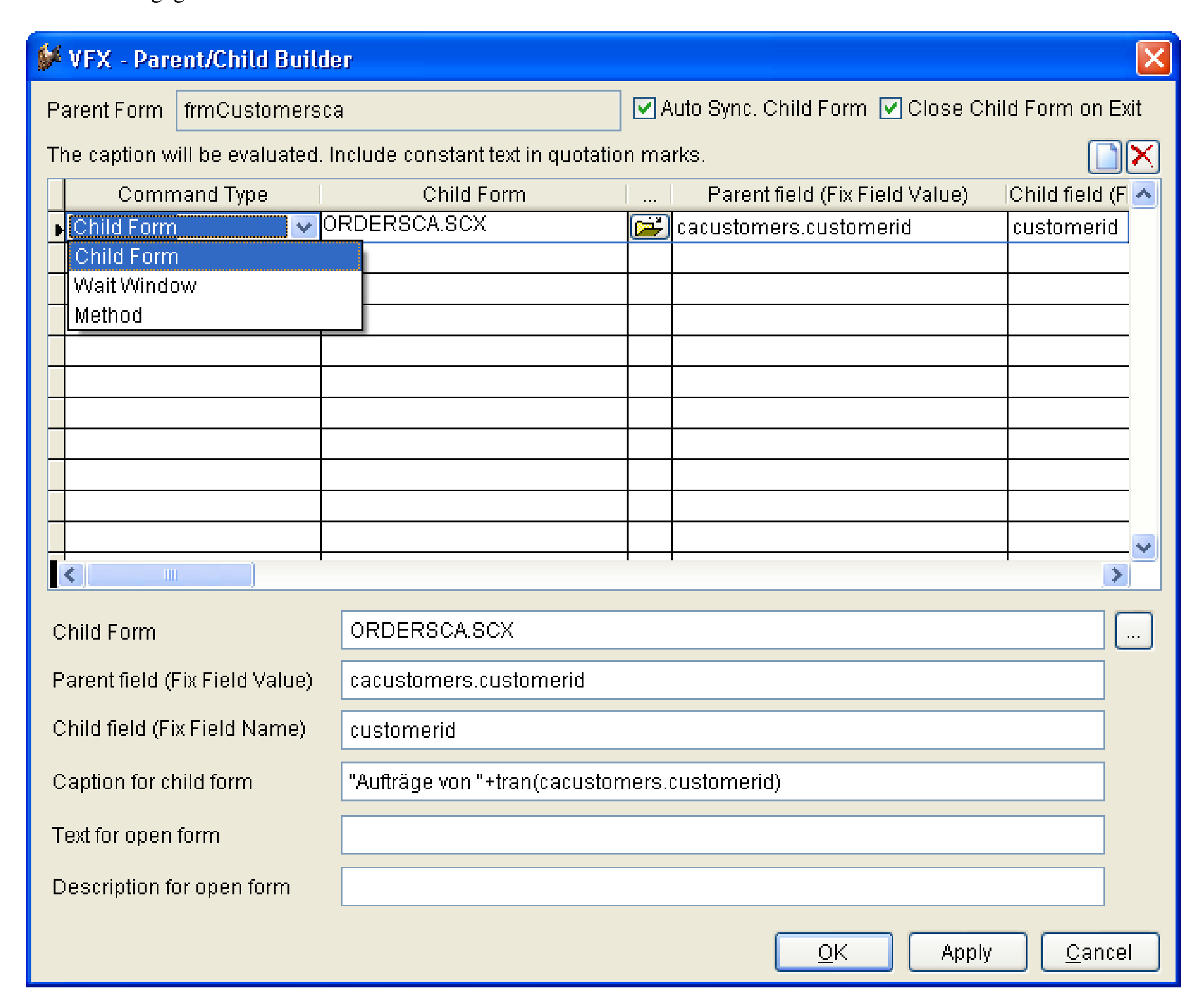
Beim erstmaligen Generieren eines neuen Formulars wird gefragt, ob das neue Formular in die Tabelle *Vfxfopen* eingetragen werden soll. Wenn diese Frage mit *Ja* beantwortet wird, kann das neue Formular über den Öffnen-Dialog gestartet werden.



19.3.8. VFX – Parent/Child Builder

In früheren VFX-Versionen war es möglich in der *OnMore*-Methode Child-Formulare oder Methoden des Parent-Formulars aufzurufen oder auch ein *Wait Window* anzuzeigen. Alle diese drei Aufrufmöglichkeiten können auch im VFX – Parent/Child Builder verwaltet werden. Dafür kann im Grid in der Spalte *Command Type* eine der drei Aufrufmöglichkeiten ausgewählt werden.

Zusätzlich zur Bearbeitungsmöglichkeit im Grid können jetzt alle Werte wahlweise auch in Textboxen unterhalb des Grids eingegeben werden.



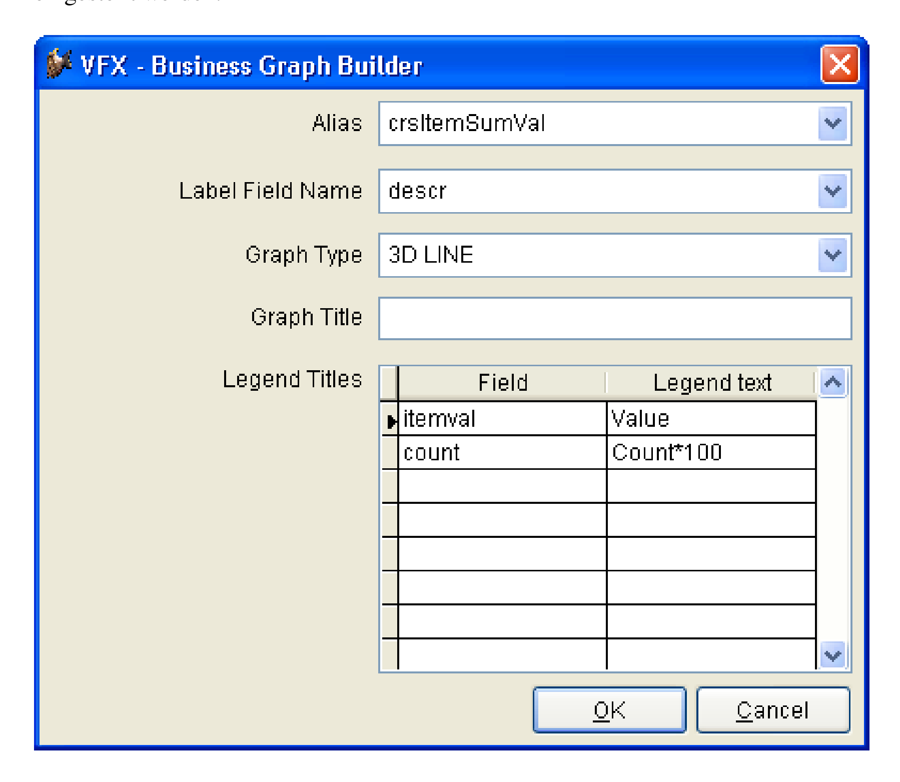
Wenn das Kontrollkästchen *Auto Sync. Child Form* markiert ist, werden beim Bewegen des Satzzeigers im Parent-Formular automatisch alle Child-Formulare synchronisiert.

Mit dem Kontrollkästchen *Close Child Form on Exit* kann eingestellt werden, ob beim Schließen des Parent-Formulars automatisch auch alle geöffneten Child-Formulare geschlossen werden sollen.

Bei der Auswahl eines Child-Formulars versucht der VFX – Parent/Child Builder möglichst die Informationen aus dem Parent- und dem Child-Formular zu ermitteln, um die folgenden Spalten im Grid richtig auszufüllen. Dabei werden auch Parent/Child-Szenarien basierend auf CursorAdaptern unterstützt. Als *Parent Field* wird dabei stets das Feld mit dem Primärschlüssel aus dem Parent-Formular vorgeschlagen.

19.3.9. VFX – Business Graph Builder

Mit dem neuen VFX – Business Graph Builder können alle Eigenschaften von *cBusinessGraph* Objekten eingestellt werden.



Eine Geschäftsgrafik wird basierend auf einem Arbeitsbereich erstellt. Dieser Arbeitsbereich kann aus der Combobox im Builder aus den Datenquellen der Datenumgebung ausgewählt oder manuell eingegeben werden. Der Arbeitsbereich wird in der Eigenschaft *cAliasName* gespeichert.

In der Combobox *Label Field Name* werden alle Felder des ausgewählten Arbeitsbereichs aufgeführt. Hier kann ein Feld ausgewählt werden, das als Beschriftung für die Datenreihen verwendet wird. Der ausgewählte Feldname wird in der Eigenschaft *cLabelField* gespeichert.

Der Darstellungstyp wird in der Combobox *Graph Type* ausgewählt. Es kann aus den Typen 3D BAR, 2D BAR, 3D LINE, 2D LINE, 3D AREA, 2D AREA, 3D STEP, 2D STEP, 3D COMBINATION, 2D PIE und 2D XY gewählt werden. Der ausgewählte Wert wird in der Eigenschaft *nGraphType* gespeichert.

In der Textbox *Graph Title* kann eine Überschrift für die Geschäftsgrafik eingegeben werden. Die Überschrift wird in der Eigenschaft *cGraphTitle* gespeichert.

Im Grid wird für jedes Feld aus dem ausgewählten Arbeitsbereich eine Zeile angezeigt, ausgenommen ist das Feld, das als Beschriftung für die Datenreihen dient. Die Daten aus allen diesen Feldern werden in der Geschäftsgrafik angezeigt. In der ersten Spalte des Grids werden die Namen der Felder angezeigt. In der zweiten Spalte sollte für jedes Feld eine Bezeichnung angegeben werden. Diese Bezeichnungen werden in der Legende der Geschäftsgrafik angezeigt. Die Legendenbezeichnungen werden in einer Komma-Separierten Liste entsprechend der Reihenfolge der Felder in der Eigenschaft *cLegendTitles* gespeichert.

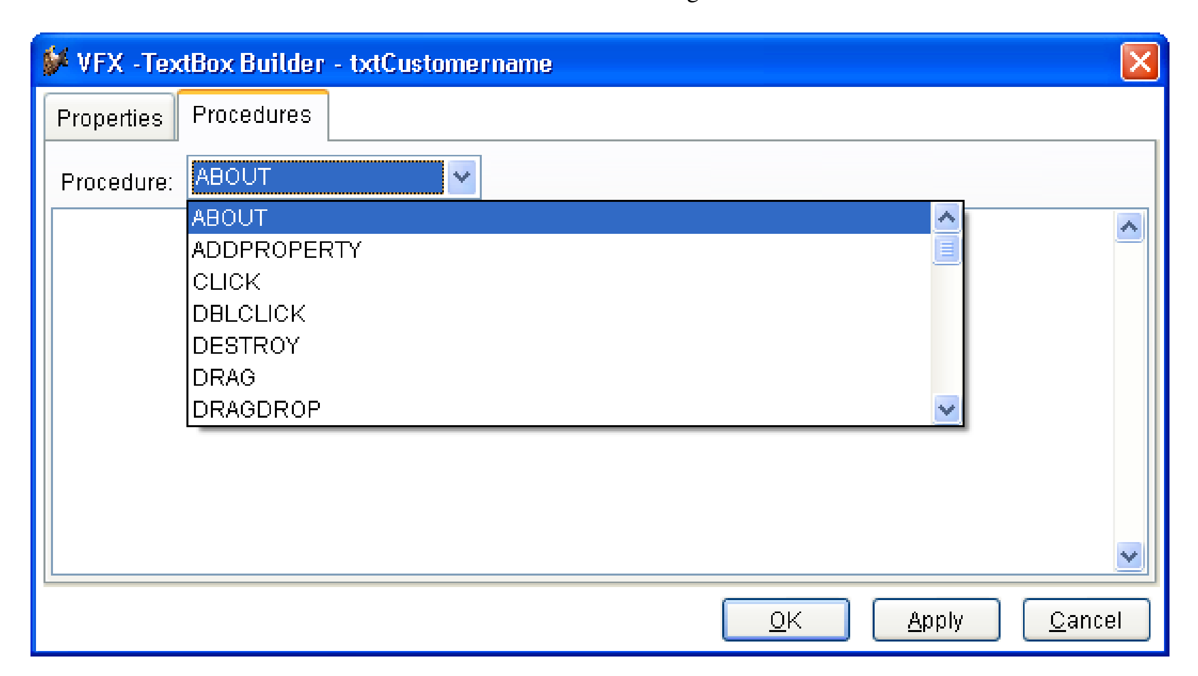
19.3.10. VFX – TextBox Builder

Im neuen VFX – TextBox Builder können die wichtigsten Eigenschaften von allen Steuerelementen basierend auf der VFP Basisklasse Textbox eingestellt werden. Auch die Bearbeitung sämtlicher Methoden ist hier möglich.

Der VFX – TextBox Builder kann aus dem VFX Menü über den Menüpunkt *VFX Power Builders* gestartet werden, wenn eine Textbox das ausgewählte Steuerelement im VFP Formular-Designer ist. Wahlweise kann der Builder auch aus dem Kontextmenü der Textbox gestartet werden.

Auf der Seite *Properties* können insbesondere die Schriftarteinstellungen gemacht werden. Im *Sample* Feld wird eine Vorschau auf die eingestellten Werte gegeben.

Auf der Seite *Procedures* kann der Code aller Methoden und Ereignisse bearbeitet werden.

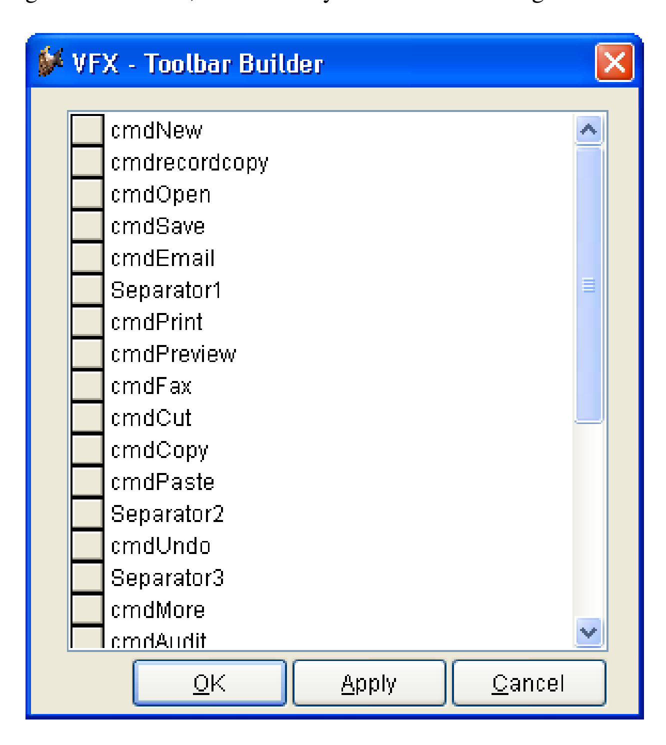


19.3.11. VFX – Toolbar Builder

Ein weiterer neuer Builder in VFX ist der Builder für Symbolleisten.

Der VFX – Toolbar Builder erleichtert es Steuerelemente und Separatoren innerhalb von Symbolleisten in eine neue Reihenfolge zu bringen. Dazu können die Elemente in einer Listbox nach oben oder unten bewegt werden.

Der VFX – Toolbar Builder kann im VFP Klassen-Designer mit der Auswahl Builder aus dem Kontextmenü gestartet werden, wenn eine Symbolleisten-Klasse geöffnet ist.



Die Anordnung der Elemente in der Listbox entspricht der Reihenfolge der Elemente in der Symbolleiste. Um die Reihenfolge zu ändern, können die Einträge mit den Mover-Bars verschoben werden.

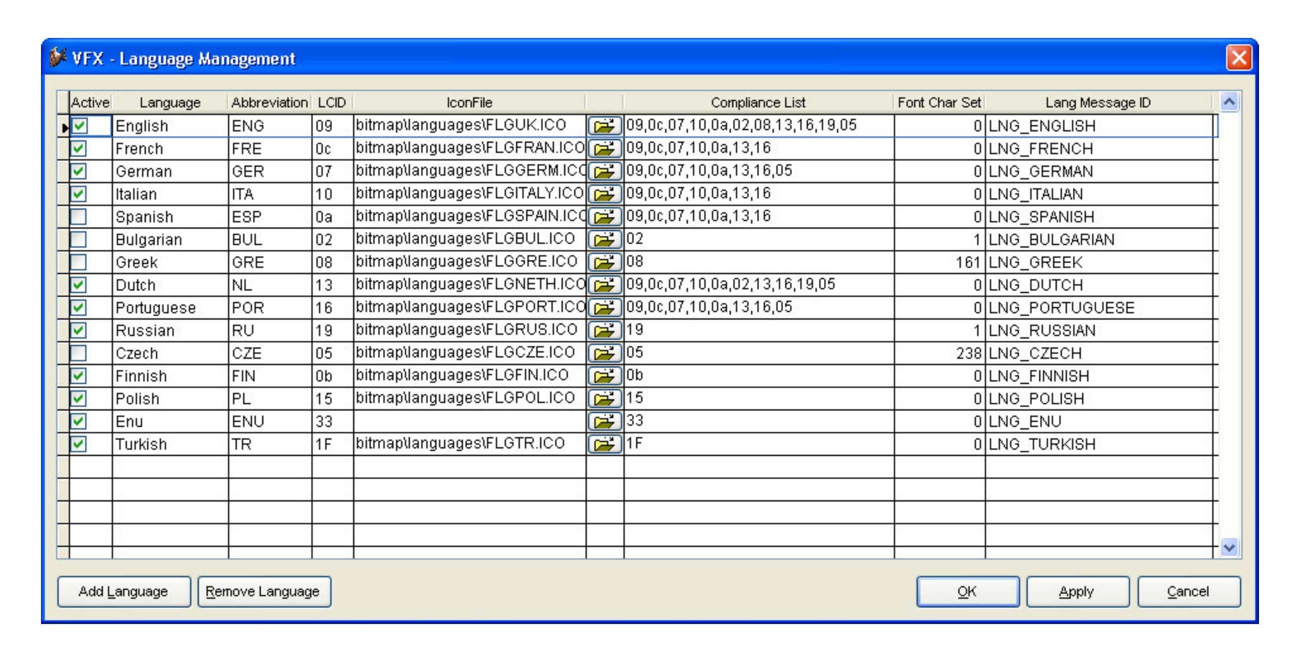
19.4. Lokalisierung

Entsprechend der gewählten Sprache werden jetzt auch die Einstellungen für das Datumsformat und das Zeitformat angepasst. Um diese Lokalisierung zu aktivieren, muss im VFX – Application Builder das *Date format* auf *VFX LOCALIZED* eingestellt werden. Wahlweise kann manuell in der Klasse *cFoxAppl* der Eigenschaft *cDateFormat* der Wert *VFX LOCALIZED* zugewiesen werden.

19.4.1. VFX - Language Management Builder

Dieser Builder verwaltet die verfügbaren Sprachen für eine Anwendung, die mit Lokalisierung zur Laufzeit arbeitet. Die Informationen über die zur Verfügung stehenden Sprachen sind in *VFXLanguage.dbf* gespeichert.

Bei der Generierung einer neuen Anwendung werden alle Sprachen, die mit VFX geliefert werden, in das neue Projekt kopiert. Hier können Sprachen hinzugefügt, aber auch gelöscht werden. Die mit VFX gelieferten Sprachen können jedoch nicht gelöscht werden, sondern nur als nicht aktiv gekennzeichnet werden. Als nicht aktiv gekennzeichnete Sprachen stehen zur Laufzeit der Anwendung nicht in den Comboboxen zur Sprachauswahl im Anmeldedialog und in der Standard-Symbolleiste zur Verfügung.



Active	Language	Abbreviation	LCID	IconFile		Compliance List	Font Char Set	Lang Message ID
✓	English	ENG	09	bitmap\languages\FLGUK.ICO		09,0c,07,10,0a,02,08,13,16,19,05	0	LNG_ENGLISH
✓	French	FRE	0c	bitmap\languages\FLGFRAN.ICO		09,0c,07,10,0a,13,16	0	LNG_FRENCH
✓	German	GER	07	bitmap\languages\FLGGERM.ICO		09,0c,07,10,0a,13,16,05	0	LNG_GERMAN
✓	Italian	ITA	10	bitmap\languages\FLGITALY.ICO		09,0c,07,10,0a,13,16	0	LNG_ITALIAN
	Spanish	ESP	0a	bitmap\languages\FLGSPAIN.ICO		09,0c,07,10,0a,13,16	0	LNG_SPANISH
	Bulgarian	BUL	02	bitmap\languages\FLGBUL.ICO		02	1	LNG_BULGARIAN
	Greek	GRE	08	bitmap\languages\FLGGRE.ICO		08	161	LNG_GREEK
✓	Dutch	NL	13	bitmap\languages\FLGNETH.ICO		09,0c,07,10,0a,02,13,16,19,05	0	LNG_DUTCH
✓	Portuguese	POR	16	bitmap\languages\FLGPORT.ICO		09,0c,07,10,0a,13,16,05	0	LNG_PORTUGUESE
✓	Russian	RU	19	bitmap\languages\FLGRUS.ICO		19	1	LNG_RUSSIAN
	Czech	CZE	05	bitmap\languages\FLGCZE.ICO		05	238	LNG_CZECH
✓	Finnish	FIN	0b	bitmap\languages\FLGFIN.ICO		0b	0	LNG_FINNISH
✓	Polish	PL	15	bitmap\languages\FLGPOL.ICO		15	0	LNG_POLISH
✓	Enu	ENU	33			33	0	LNG_ENU
✓	Turkish	TR	1F	bitmap\languages\FLGTR.ICO		1F	0	LNG_TURKISH

In der Spalte *Language* wird die Bezeichnung einer Sprache eingetragen, so wie sie in der Sprachauswahl-Combobox zur Laufzeit angezeigt werden soll.

In der Spalte *Abbreviation* wird der Name des Feldes in der Tabelle *Vfxmsg.dbf* eingetragen. Aus diesem Feld werden zur Laufzeit die Texte der gewählten Sprache gelesen.

Die Spalte *LCID* enthält den *Locale Identifier* der Sprache. Dies ist ein in Windows definierter Wert und wird für die Regionaleinstellungen verwendet.

In der Spalte *IconFile* kann der Name zu einer Icondatei ausgewählt werden. Das Icon sollte eine Flagge zur Veranschaulichung einer Sprache anzeigen. Das Icon wird in der Sprachauswahl-Combobox zur Laufzeit angezeigt.

Die Spalte *Compliance List* enthält eine durch Komma separierte Liste *Locale Identifiers* von Sprachen, die zur aktuellen Sprache kompatibel sind. Diese Liste enthält Werte von Regionaleinstellungen, die geeignet sind, die aktuelle Sprache korrekt anzuzeigen.

Die Spalte *Lang Message Id* enthält den Wert des Feldes *MessageID* aus der Tabelle *Vfxmsg.dbf,* die die lokalisierten Texte der aktuellen Sprache für die Sprachauswahl-Combobox enthält.

Die oben beschriebenen Einstellungen können für mit VFX gelieferte Sprachen nicht geändert werden. Diese Einstellungen können nur für neu hinzugefügte Sprachen bearbeitet werden. Eine neue Sprache kann über die Schaltfläche *Add Language* hinzugefügt werden. Eine so hinzugefügte Sprache wird in der Sprachauswahl-Combobox angezeigt. Für eine neu angelegte Sprache wird automatisch ein Feld in der Tabelle *Vfxmsg.dbf* mit Bezeichnung aus der Spalte *Abbreviation* angelegt.

Um eine neu hinzugefügte Sprache verwenden zu können, müssen alle Texte aus der Tabelle *Vfxmsg.dbf* in die neue Sprache übersetzt werden.

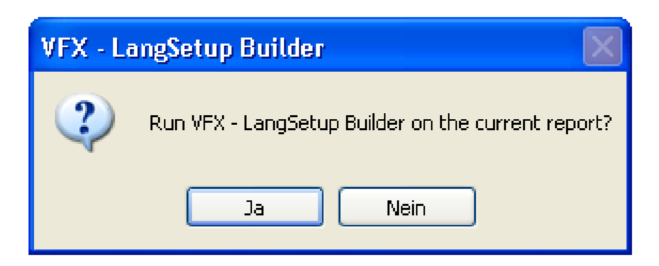
Ausführen der Methode LangSetup

Bei Aufruf einer *LangSetup*-Methode auf Formularebene werden automatisch alle auf dem Formular befindlichen Objekte nach Vorhandensein einer *LangSetup*-Methode durchsucht. Container-Objekte werden rekursiv durchsucht. Die *LangSetup*-Methode wird so in allen Objekten ausgeführt.

19.4.2. VFX - LangSetup Builder

Der VFX – LangSetup Builder unterstützt nicht nur Formulare, sondern auch Berichte sowie den Öffnen-Dialog.

Um den VFX – LangSetup Builder auf einer Berichtsdatei anzuwenden, muss die Berichtsdatei zunächst im VFP Berichts-Designer geöffnet werden. Jetzt kann der VFX – LangSetup Builder aus dem VFX-Menü gestartet werden. Es erscheint folgende Messagebox:



Wenn mit „ja" geantwortet wird, fügt der VFX – LangSetup Builder für jede Bezeichnung in dem Bericht einen Datensatz an die Tabelle *Vfxmsg.dbf* an. Die Bezeichnungen im Bericht werden durch „^" gefolgt von der *Message_Id* des neuen Datensatzes in *Vfxmsg.dbf* ersetzt. Zur Laufzeit ersetzt ein ReportListener die Bezeichnungen im Bericht durch die lokalisierten Texte aus der Tabelle *Vfxmsg.dbf*.

Wenn nur ein Projekt, aber kein Formular oder Bericht im Designer geöffnet ist und der VFX – LangSetup Builder gestartet wird, wird die Tabelle *Vfxfopen.dbf* lokalisiert.

19.5. Project Hook

Bereits in VFX 9.0 wurden zur Laufzeit Objekte angelegt für alle Felder aus *Vfxsys.dbf* und aus dem aktuellen Datensatz aus *Vfxusr.dbf*. Aus Gründen der Kompatibilität mit früheren VFX-Versionen gibt es die Datei *Vfxglobal.h*, die Ersetzungen für alle Variablennamen enthält, die in früheren VFX-Versionen verwendet wurden. Die Datei *Vfxglobal.h* wird automatisch bei jeder Neuerstellung des Projekts oder beim Erstellen einer App- oder Exe-Datei neu angelegt. Dabei werden alle Felder, also auch selbst hinzugefügte, aus den Dateien *Vfxsys.dbf* und *Vfxusr.dbf* berücksichtigt.

19.6. Produktaktivierung

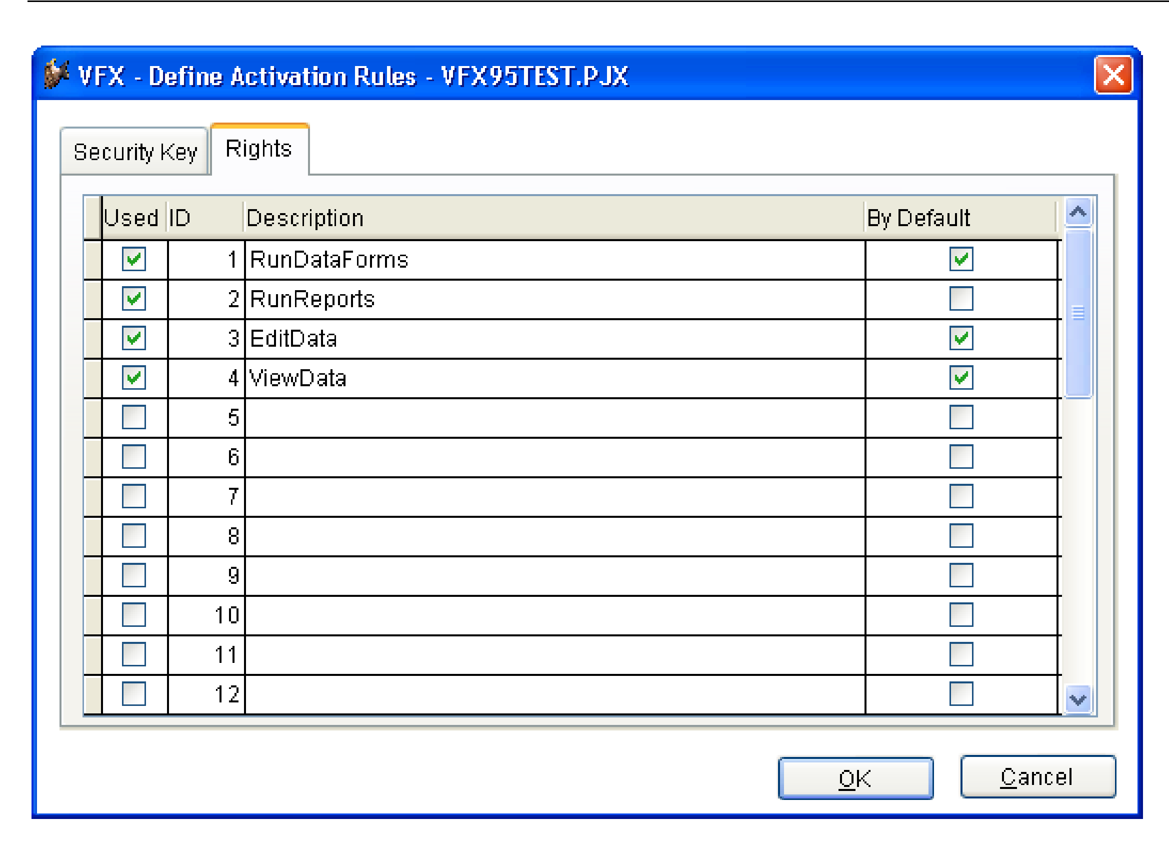
VFX Anwendungen können vor unberechtigter Nutzung durch eine Produktaktivierung geschützt werden.

Für die in VFX integrierte Produktaktivierung steht jetzt ein Web Service zur Verfügung. Ähnlich wie bei der Aktivierung von VFX können sich Benutzer jetzt über einen Web Service einen Aktivierungsschlüssel für eine Anwendung holen.

Zur Verwaltung der Aktivierungsschlüssel und Kundendaten steht die neue Anwendung VFX – Kundenverwaltung zur Verfügung.

19.6.1. Definieren der Aktivierungsregeln

Zunächst müssen jedoch in der Anwendung die Aktivierungsregeln festgelegt werden. Dies geschieht im Dialog *VFX – Define Activation Rules*. Die Definition der Regeln geschieht genau so, wie in VFX 9.0. Die Aktivierungsregeln werden in der Klassenbibliothek *Appl.vcx* in der Klasse *cVfxActivation* in der Eigenschaft *cActPattern* verschlüsselt gespeichert.



Um ein Recht zu aktivieren muss zunächst die Checkbox in der ersten Grid-Spalte markiert werden. In der Spalte *Description* muss dem Recht ein Name gegeben werden. Zur Laufzeit wird dem Objekt *SecurityRights* eine Eigenschaft mit dem gewählten Namen hinzugefügt.

In der Spalte *By Default* kann eingestellt werden, ob dieses Recht standardmäßig aktiviert werden soll. Die Standardwerte gelten für neu angelegte Benutzer, können in der Kundenverwaltung je Benutzer geändert werden.

Die Registrierungsnummer ist ein numerischer Wert mit 10 Stellen Länge. Der Benutzer muss die Registrierungsnummer dem Entwickler mitteilen oder per E-Mail senden. Der Entwickler erfasst in der VFX – Kundenverwaltung einen neuen Datensatz für diesen Benutzer und gibt hier die Registrierungsnummer ein.

19.6.2. Aktivierungsschlüssel erstellen

Der Aktivierungsschlüssel enthält die Berechtigungen, für die einzelnen Module der Anwendung. Aktivierungsschlüssel für VFX Anwendungen werden mit der VFX – Kundenverwaltung erstellt. Die VFX – Kundenverwaltung ist ein eigenständiges Projekt und wird mit VFX geliefert. Aus dem Projekt VFX – Kundenverwaltung kann eine Exe-Datei erstellt werden. Die Verwaltung der Kundendaten und die Generierung von Aktivierungsschlüsseln kann so auf einem PC unabhängig vom Entwicklungsrechner durchgeführt werden.

Damit die VFX – Kundenverwaltung Aktivierungsschlüssel erstellen kann, müssen ihr die Aktivierungsregeln bekannt sein. Die Aktivierungsregeln sind aber in der eigentlichen Anwendung in der Klassenbibliothek *Appl.vcx* gespeichert.

Bei der Generierung eines Aktivierungsschlüssels benutzt die VFX – Kundenverwaltung die Registrierungs-Dll. Das Projekt zur Erstellung der Registrierungs-Dll befindet sich unterhalb des Projektordners der Anwendung und wird vom VFX – Application Wizard in jedes neue Projekt kopiert. Der Name des Registrierungsprojekts ist: *„Register" + <Name der Anwendung>*

Bei der Erstellung der Registrierungs-Dll werden über einen Projekt Hook die Aktivierungsregeln aus der Klassenbibliothek *Appl.vcx* der Anwendung gelesen und in der Klasse *cRegDll* in der Eigenschaft *cActPatternName* gespeichert. Die Registrierungs-Dll enthält also die Aktivierungsregeln. Dadurch ist die VFX – Kundenverwaltung unabhängig von einer bestimmten Anwendung. Die VFX – Kundenverwaltung kann Registrierungs-Dlls für verschiedene Anwendungen benutzen. Die Registrierungs-Dll kann aus dem VFX-Menü über den Menüpunkt *Activation, Build register DLL* erstellt werden.

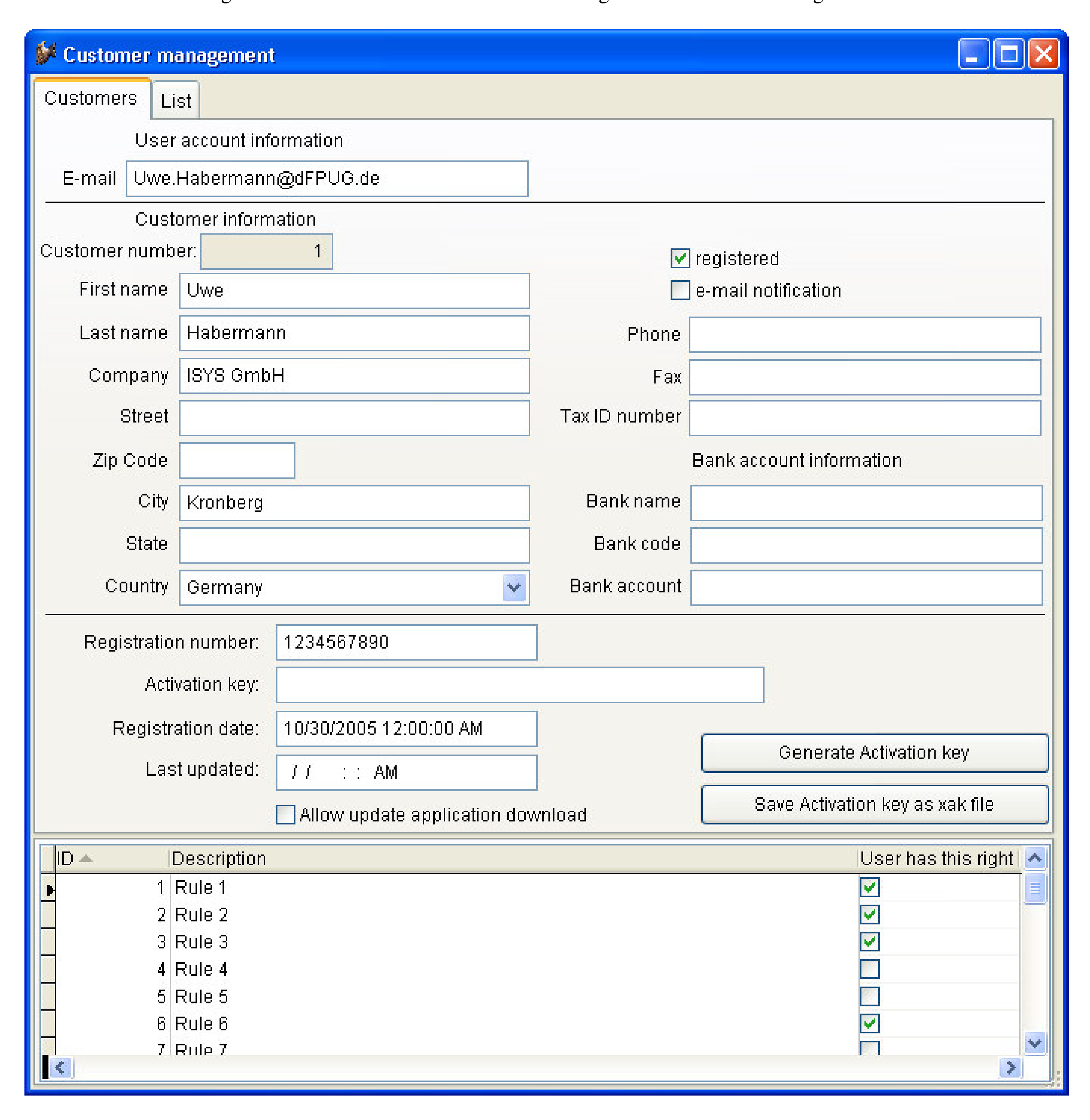
In der Registrierungs-Dll befindet sich die Methode *generateactkey*. Die Parameter dieser Methode sind die Registrierungsnummer, eine Zeichenkette mit den zu erteilenden Rechten sowie der Pfad zur Datei *VFXGenActKey.app*. Der Rückgabewert ist der generierte Aktivierungsschlüssel.

Die Registrierungs-Dll ruft eine Funktion der Anwendung *VFXGenActKey.app* auf. *VFXGenActKey.app* wird mit VFX geliefert und befindet sich im Projektordner der VFX – Kundenverwaltung. *VFXGenActKey.app* enthält den Algorithmus, der anhand der Aktivierungsregeln einen Aktivierungsschlüssel erstellt. Der Quell-Code von *VFXGenActKey.app* wird nicht mit VFX geliefert. Entwickler, die den Quell-Code zur Verfügung hätten, könnten Aktivierungsschlüssel für VFX-Anwendungen anderer Entwickler erstellen.

Die VFX – Kundenverwaltung greift auf die Kundendatenbank über die Datei *Config.vfx* zu. Die Kundendatenbank kann sich in einer VFP-Datenbank oder in einer SQL Server-Datenbank befinden. Der Datenzugriff erfolgt genauso, wie bei anderen VFX-Anwendungen auch. Für die VFX – Kundenverwaltung wurde der Datei *Config.vfx* eine Spalte hinzugefügt. Die Spalte *RegDllName* enthält den Namen der zu verwendenden Registrierungs-Dll.

19.6.3. VFX – Kundenverwaltung

Diese VFX-Anwendung enthält zwei Formulare: Kundenverwaltung und Versionsverwaltung.



Im Formular Kundenverwaltung können die Kundendaten bearbeitet werden. Zu jedem Kunden werden die Registrierungsnummer und die vergebenen Rechte gespeichert. Die Rechte können bei Bedarf verändert werden und es kann aus diesem Formular ein neuer Aktivierungsschlüssel generiert werden.

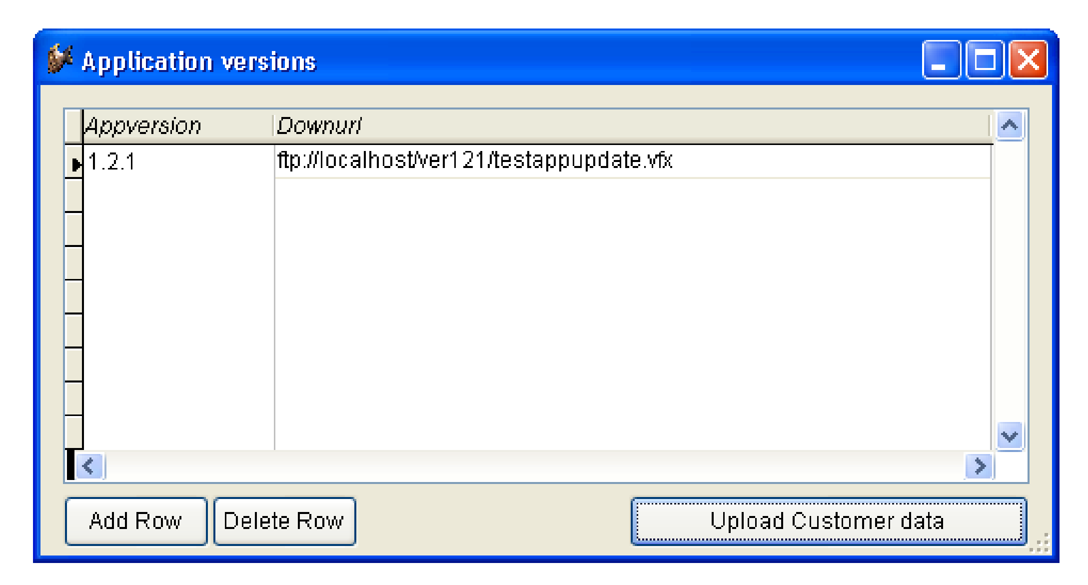
Über die Schaltfläche *Generate Activation Key* kann ein Aktivierungsschlüssel entsprechend der eingegebenen Registrierungsnummer und Benutzerrechte erstellt werden. Der generierte Aktivierungsschlüssel wird in der Kundenverwaltung gespeichert.

Über die Schaltfläche *Save Activation key as xak file* kann der angezeigte Aktivierungsschlüssel in einer Datei mit dem Namen <*Projektname*>*.xak* im aktuellen Ordner gespeichert werden. Diese Datei kann an den Kunde

geschickt und im Ordner der Anwendung gespeichert werden. Beim Start der Anwendung wird der Aktivierungsschlüssel automatisch aus der Datei gelesen und die Anwendung wird damit aktiviert.

Aus der Kundenverwaltung können E-Mails mit der XAK-Datei als Anhang versendet werden. Diese E-Mails werden versendet, wenn der Registrierungstyp (*cvfxActivation.nRegWay*) der Anwendung ungleich 10 ist. Beim Registrierungstyp 10 wird erwartet, dass sich der Anwender den Aktivierungsschlüssel über einen Web Service holt. In diesem Fall bekommt der Anwender eine E-Mail mit einer Anleitung zum Erhalt des Aktivierungsschlüssels. Der Betreff und Text der generierten E-Mails können im Formular *Manage E-Mail Texts* bearbeitet werden. Der E-Mailtext wird mit der *Textmerge* Funktion von VFP verarbeitet und kann so beliebige Felder aus der Kundenverwaltung enthalten. Auf diesem Weg können persönliche E-Mails gestaltet werden.

Mit dem Formular Versionsverwaltung werden die Versionen und Download-Links der Anwendung verwaltet. Neue Versionen der Anwendung können auf einem Internet Server bereitgestellt werden. Wenn ein Kunde seine Anwendung aktualisieren will, lädt die Anwendung zunächst die Datei *UpdateCustomers.vfx* herunter. In dieser Datei befinden sich die Registrierungsnummern, der zur Aktualisierung berechtigten Kunden. Wenn die Aktualisierungsberechtigung besteht, wird die Datei *Updateversions.vfx* heruntergeladen. In dieser Datei befinden sich die Download-Links zu den verfügbaren Anwendungsversionen. Die Download-Links zu den Dateien *UpdateCustomers.vfx* und *UpdateVersions.vfx* sind in der Anwendung gespeichert und können mit dem VFX – Application Builder eingestellt werden.



Durch einen Mausklick auf die Schaltfläche *Upload Customer Data* werden die Dateien *UpdateCustomers.vfx* und *UpdateVersions.vfx* erstellt.

Damit diese Dateien auf den Internet Server hochgeladen werden können, müssen die Anmeldeinformationen im Projekt der Registrierungs-Dll in der Klassenbibliothek *Regdll.vcx* in der Klasse *cRegDll* gespeichert werden.

Eigenschaften

cFtpUrl – Domainname eines FTP-Servers.

cFtpDir – Verzeichnis auf dem FTP-Server, in dem die Dateien *UpdateCustomers.vfx* und *UpdateVersions.vfx* gespeichert werden sollen.

cPort – Zu verwendender Übertragungsport. Standardmäßig wird für FTP-Uploads der Port 21 verwendet.

cUserName – Benutzername für den FTP-Zugang.

cPassword – Kennwort für den FTP-Zugang.

19.6.4. Web Service für die Registrierung

Mit der VFX – Kundenverwaltung wird das Projekt *RegistrationWebService.pjx* im Ordner *RegistrationWebService* unterhalb der VFX – Kundenverwaltung geliefert. Aus diesem Projekt kann ein COM Server erstellt werden, der auf einem Internet Information Server als Web Service installiert werden kann. An diesen Web Service können Anwendungen die Registrierungsdaten von Benutzern senden. Der Web Service kann an die Anwendung einen Aktivierungsschlüssel senden.

Der Web Service benutzt eine *Config.vfx* für den Datenzugriff. Hier muss auf die gleiche Datenbank gezeigt werden, die auch von der VFX – Kundenverwaltung verwendete wird. Im einfachsten Fall kann der Web Service im gleichen Ordner wie die VFX – Kundenverwaltung installiert werden und so die gleiche *Config.vfx* benutzen, wie die Kundenverwaltung.

Wenn der Web Service auf einem entfernten Internet Server laufen soll, so ist die Kundendatenbank mit dem VFX – Upsizing Wizard auf einen SQL Server zu portieren. Auf die so erzeugte SQL Server Datenbank können sowohl der Web Service, als auch die VFX – Kundenverwaltung über das Internet zugreifen.

Damit eine Anwendung über den Web Service aktiviert werden kann, müssen in der Anwendung ein paar Einstellungen im Formular *VfxRegister.scx* gemacht werden:

cWSDL – Enthält die URL der WSDL Datei. Diese Datei wird bei der Registrierung des Web Service mit dem SOAP Toolkit auf dem Internet Server generiert.

cServiceName – Enthält den Namen des Web Service.

cServiceMethodName – Enthält den Namen der verwendeten Web Service Methode. Standardmäßig ist dies die Methode *GenerateActKey*.

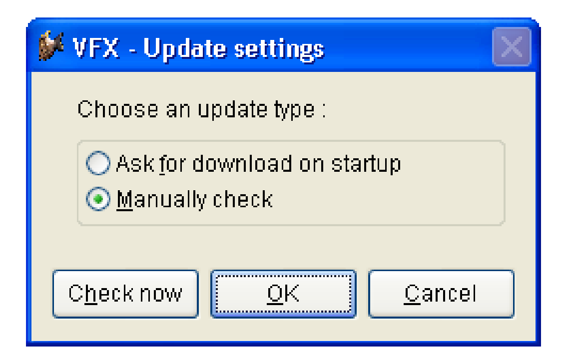
Wenn sich ein Kunde über den Web Service registriert, werden die Registrierungsdaten an den Web Service im XML Format übertragen. Der Web Service sucht in der Kundendatenbank nach einem Datensatz mit der gleichen E-Mailadresse sowie dem gleichen Anwendungsnamen und der gleichen Version. Wenn dieser Datensatz gefunden wird, wird der dort eingetragene Aktivierungsschlüssel an den Kunden übertragen. Die Anwendung wird dabei automatisch aktiviert. Wenn kein solcher Datensatz gefunden wird, werden die Registrierungsinformationen in der Kundendatenbank gespeichert.

Der Web Service für die Registrierung enthält die Methode *ReceiveErrorInfo*. Diese Methode kann Fehlermeldungen von Endanwendungen empfangen. Damit haben Anwender die Möglichkeit die Dateien, die lokal in der Fehlerprotokolltabelle *Vfxlog* gespeichert werden, an den Web Service zu senden.

19.7. Aktualisierung von VFX

Es ist sinnvoll VFX regelmäßig zu aktualisieren, damit immer der aktuelle Stand zur Verfügung steht. VFX kann automatisch auf Aktualisierungen prüfen. Dies kann im Dialog *Update Settings* eingestellt werden. Wenn die Option *Ask for download on startup* gewählt ist, überprüft VFX bei jedem ersten Start an jedem Tag, ob ein aktualisierter Build zur Verfügung steht. Falls ja, wird gefragt, ob der neue Build heruntergeladen und installiert werden soll. Die Überprüfung wird nicht durchgeführt, wenn keine Verbindung mit dem Internet besteht. Über

die Schaltfläche *Check for updates now* kann die Überprüfung nach aktualisierten Builds jederzeit manuell gestartet werden.



19.7.1. Die Klasse cUpdate

Diese Klasse stellt den Dialog zur Bearbeitung der Aktualisierungseinstellungen bereit. Über eine Schaltfläche kann sofort geprüft werden ob Aktualisierungen zur Verfügung stehen. Die Klasse wird sowohl von VFX intern als auch in Endanwendungen verwendet.

tlUpdateApp - Initialisierungsparameter.
.T. – Aktualisierung für Endanwendungen.
.F. – Aktualisierung für VFX.

Methode

WriteData()
Speichert die gewählte Methode zur Aktualisierung in der Tabelle *Vfxsys*.

19.7.2. Die Klasse cUpdateEngine

Mit dieser Klasse wird die Aktualisierung der Anwendung beim Kunden durchgeführt. Abhängig von den Einstellungen im Dialog für die Aktualisierungseinstellungen wird eine Datei heruntergeladen, die die aktuell verfügbare Versionsnummer enthält. Um eine neue Programmversion herunterzuladen wird ein Objekt der Klasse *cDownload* instanziiert.

Eigenschaften

cIniURL – URL der Datei, die die aktuelle Versionsnummer enthält.

cVersion – Versionsnummer der aktuell laufenden Anwendung.

dLastChecked – Datum der letzten Überprüfung auf Aktualisierungen. Dieses Datum wird aus der Tabelle *Vfxsys* gelesen.

nManualCheck – Wie wurde die Überprüfung auf Aktualisierungen gestartet?

- 0 – Automatische Überprüfung auf Aktualisierungen bei Anwendungsstart.
- 1 – Manuelle Überprüfung auf Aktualisierungen aus dem Menü.

nUpdateType – Aktualisierungstyp. Dieser Wert wird aus der Tabelle *Vfxsys* gelesen.

- 1 – Automatisches herunterladen und installieren, wenn Aktualisierungen vorhanden sind. Die Überprüfung wird beim ersten Anwendungsstart täglich durchgeführt.
- 2 – Automatisches herunterladen von Aktualisierungen mit anschließender Frage, ob die Aktualisierung installiert werden soll. Die Überprüfung wird beim ersten Anwendungsstart täglich durchgeführt.
- 3 – Überprüfung nach verfügbaren Aktualisierungen täglich beim ersten Anwendungsstart. Bei vorhandener aktualisierter Programmversion wird der Benutzer gefragt, ob er die Aktualisierung herunterladen und installieren will.
- 4 – Manuelle Überprüfung. Die Überprüfung auf Aktualisierungen kann aus dem Menü aufgerufen werden.

Methoden

CheckUpdType()

Ermittelt die Versionsnummer der laufenden Anwendung sowie den Aktualisierungstyp und das Datum der letzten Überprüfung auf Aktualisierung. Die ermittelten Werte werden in den Eigenschaften *cVersion, nUpdateType* und *dLastChecked* gespeichert. Wenn alle Werte erfolgreich ermittelt werden können, wird .T. zurückgegeben. Bei Auftreten eines Fehlers, wird .F. zurückgegeben.

DoUpdate(tnUpdateType)

tnUpdateType = nUpdateType

Aufruf der Methode *UpdateApp(tnUpdateType)* (in *Vfxfunc.prg*).

NewVersion()

Herunterladen der Datei aus *cIniURL.* Rückgabewert:

0 – Es ist keine neue Version verfügbar.
1 – Eine neue Version ist verfügbar.
2 – Es ist ein Fehler aufgetreten.

SetNewCheckDate(tdDate)
tdDate – Datum, an dem die nächste Überprüfung auf Aktualisierungen stattfindet. Das als Parameter übergebene Datum wird in der Tabelle *Vfxsys* im Feld *lastcheckd* gespeichert.

StartUpdate(nManualCheck)
nManualCheck – Einstellen des Aktualisierungstyps.
0/.F. – Automatische Aktualisierung.
1/.T. – Manuelle Aktualisierung.
Dies ist die wichtigste Methode dieser Klasse. Hiermit wird die Aktualisierung gestartet. Bevor diese Methode aufgerufen wird, muss der Eigenschaft *cIniURL* die URL mit der Versionsdatei zugewiesen werden. Aus dieser Methode werden die Methoden *CheckUpdType()* und *NewVersion()* aufgerufen. Die Methode *SetNewCheckDate(tdDate)* wird aufgerufen, um das Datum der letzten Überprüfung zu speichern. Wenn eine neue Version verfügbar ist, wird die Methode *DoUpdate(tnUpdateType)* aufgerufen. Bei manueller Überprüfung auf Aktualisierungen wird die Methode *DoUpdate(tnUpdateType)* ohne vorherige Prüfung aufgerufen.

In der Klassenbibliothek *Appl.vcx* befindet sich die Klasse *cAppUpdateEngine*. Dies ist eine 1:1-Ableitung der Klasse *cUpdateEngine*. In der Klasse *cAppUpdateEngine* sollten Entwickler eigene Einstellungen machen. Die Eigenschaft *cIniURL* kann auch im VFX – Application Builder eingestellt werden.

19.8. AFX Unterstützung

In der Klasse cFoxAppl befinden sich zwei neue Methoden, die die Generierung von Web-Anwendungen basierend auf VFX-Projekten vereinfachen. Es sind die Methoden *VFXMessageBox* und *VFXWaitWindow*. Diese Methoden akzeptieren die gleichen Parameter wie die entsprechende VFP-Funktion *Messagebox* bzw. der Befehl *Wait Window*.

Methoden

VFXMessageBox()
LPARAMETERS *emessagetext, ndialogboxtype, ctitlebartext, ntimeout*

emessagetext – Anzuzeigender Text in der Messagebox. Wenn dieser Wert fehlt oder von falschem Typ ist, wird eine leere Zeichenkette angezeigt.
ndialogboxtype – Typ der Messagebox. Hierüber können Schaltflächen und das Icon eingestellt werden. Der Standardwert ist 0.
ctitlebartext – Titel der Messagebox. Wenn dieser Wert fehlt oder von falschem Typ ist, wird eine leere Zeichenkette angezeigt.
ntimeout – Zeitspanne zur Anzeige der Messagebox. Wenn dieser Wert fehlt oder von falschem Typ ist, wird gewartet, bis der Benutzer eine Schaltfläche betätigt.

Diese Methode führt die VFP-Funktion MESSAGEBOX() aus. Wenn die Anwendung als Web-Anwendung läuft, werden Web-Seiten angezeigt. Wenn die Anwendung ohne sichtbare Ausgaben läuft, wird der Wert der Standardschaltfläche der Messagebox zurückgegeben.

VFXWaitWindow()
LPARAMETERS *tcMessageText, tnRow, tnColumn, tlNowait, tlClear, tlNoclear, tnTimeout*

tcMessageText – Anzuzeigender Text. Der Standardwert ist eine leere Zeichenkette.

tnRow – Zeilennummer des Wait Window. Nur in Zusammenhang mit *tnColumn* verwendbar.

tnColumn – Spaltennummer des Wait Window. Nur in Zusammenhang mit *tnRow* verwendbar.

tlNowait - .T. um die Programmausführung nach Anzeige des Wait Window fortzusetzen, .F. um die Programmausführung anzuhalten bis der Benutzer eine Taste oder Maustaste drückt. Der Standardwert ist .F.

tlClear - .T. um aktuell angezeigte Wait Windows zu löschen. Der Standardwert ist .F.

tlNoclear - .T. und das Wait Window angezeigt zu lassen, bis der Befehl WAIT CLEAR ausgeführt wird. Der Standardwert ist .F.

tnTimeout – Anzeigedauer des Wait Window. Wenn 0 angegeben wird, wird das Wait Window nicht gelöscht, bis der Befehl WAIT CLEAR ausgeführt wird. Der Standardwert ist 0.

Diese Methode führt den VFP-Befehl WAIT WINDOW aus. Wenn die Anwendung ohne sichtbare Ausgaben läuft, erfolgt keine Anzeige.

19.9. Hilfe

In der neuen Hilfe-Sektion des VFX-Menüs kann das Benutzerhandbuch geöffnet werden und es werden nützliche Informationen rund um VFX sowie Links zu Online Ressourcen angeboten.

19.9.1. Benutzerhandbuch und Dokumentation der Neuheiten

Über den Menüpunkt *VFX Help, User Manuals and What's New* können das Benutzerhandbuch und die Neuheitendokumentation in deutscher und englischer Sprache geöffnet werden. Alle diese Dokumente liegen als PDF-Dateien vor. Zum Öffnen ist der Adobe Reader erforderlich.

Über den Menüpunkt *Update Notes* kann ein Dokument geöffnet werden, das alle Änderungen in der VFX-Template-Anwendung seit dem Erscheinen beschreibt.

Wenn eins der anzuzeigenden Dokumente nicht installiert ist, wird das entsprechende Dokument automatisch heruntergeladen.

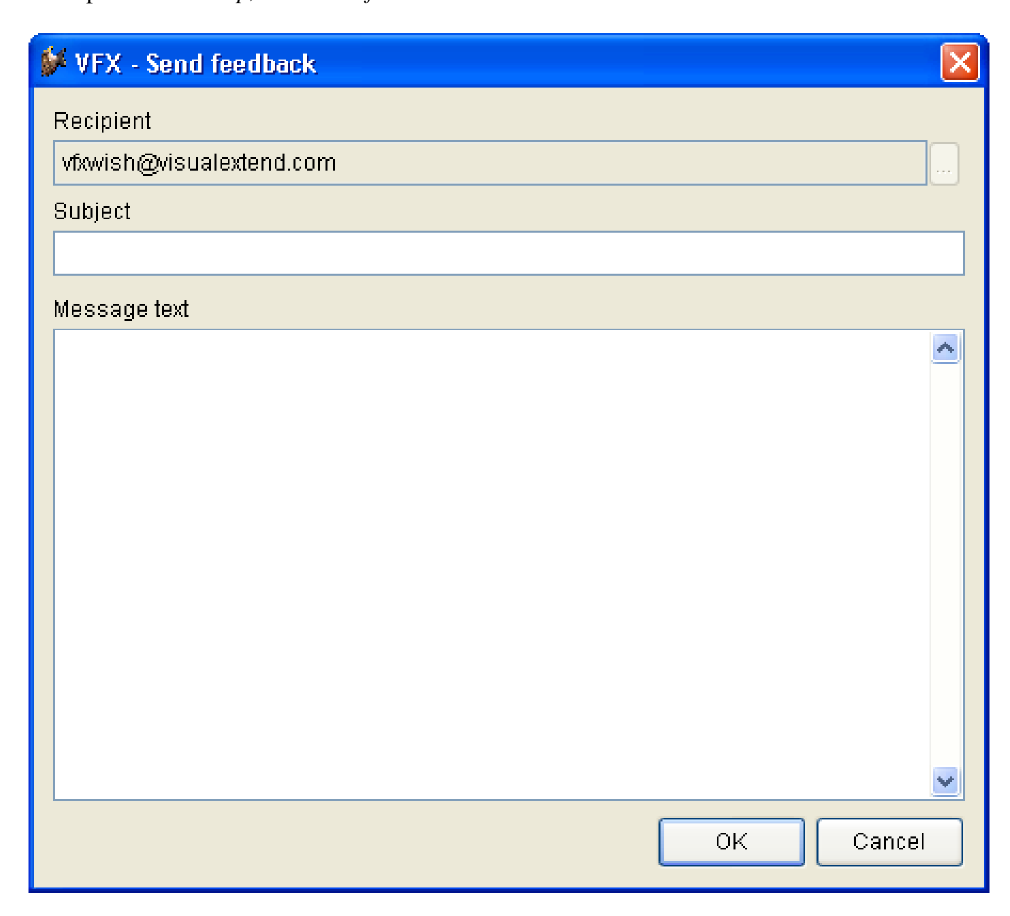
19.9.2. Visual Extend Online

Über den Menüpunkt *VFX Help, Visual Extend Online* erreichen Sie direkt die folgenden Seiten der VFX Website:

- VFX Startseite (zurzeit in vier Sprachen verfügbar - deutsch, englisch, französisch und spanisch)
- VFX Directory (Startseite des Visual Extend Portals)
- Forum (lesen und schreiben Sie im Online-Forum)
- Newsletters (abonnieren Sie aktuelle Informationen über VFX)
- DevCons (besuchen Sie die Entwicklerkonferenz!)
- Online shop (kaufen Sie VFX, Bücher, VFP und anderes)

19.9.3. Senden Sie uns eine E-Mail!

Machen Sie Vorschläge, sagen Sie uns Ihre Meinung oder äußern Sie Erweiterungswünsche. Über den Menüpunkt *VFX Help, Send user feedback* können Sie dem VFX-Team eine E-Mail senden.

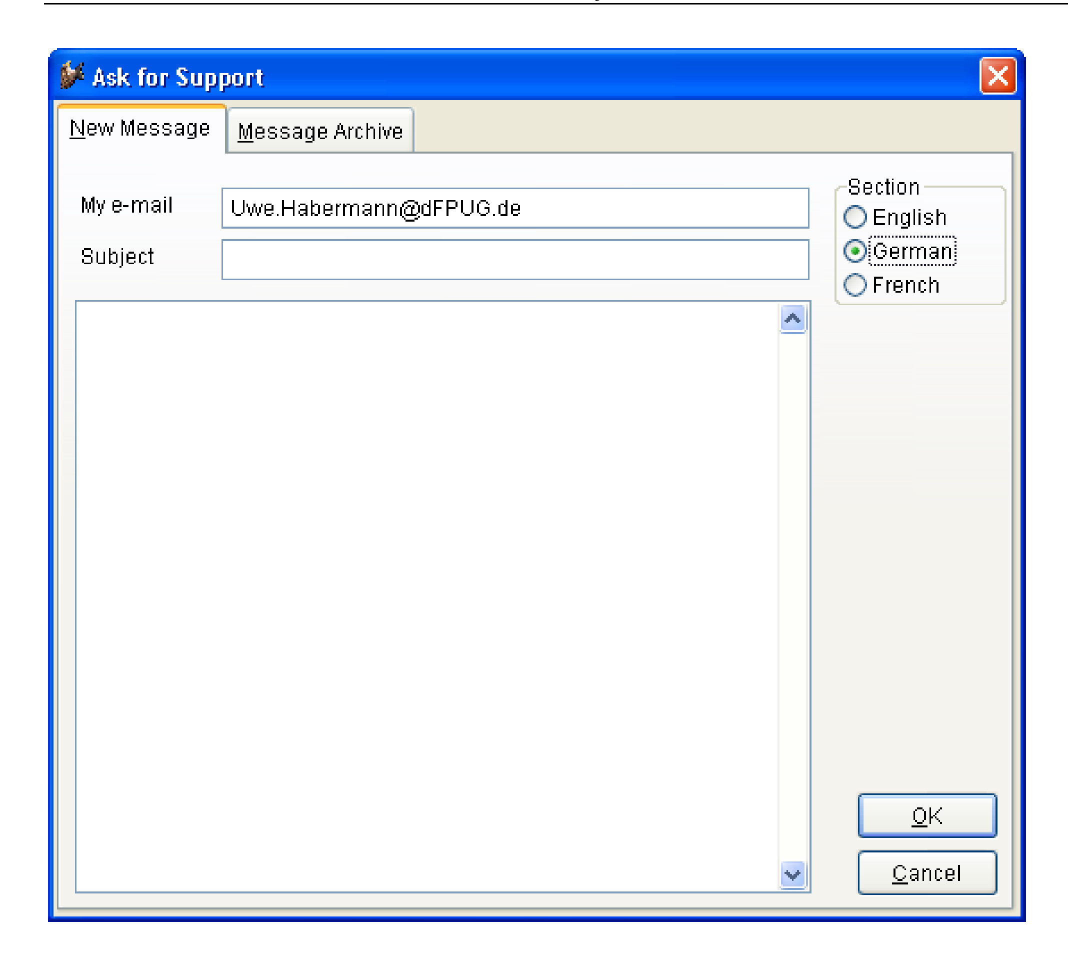


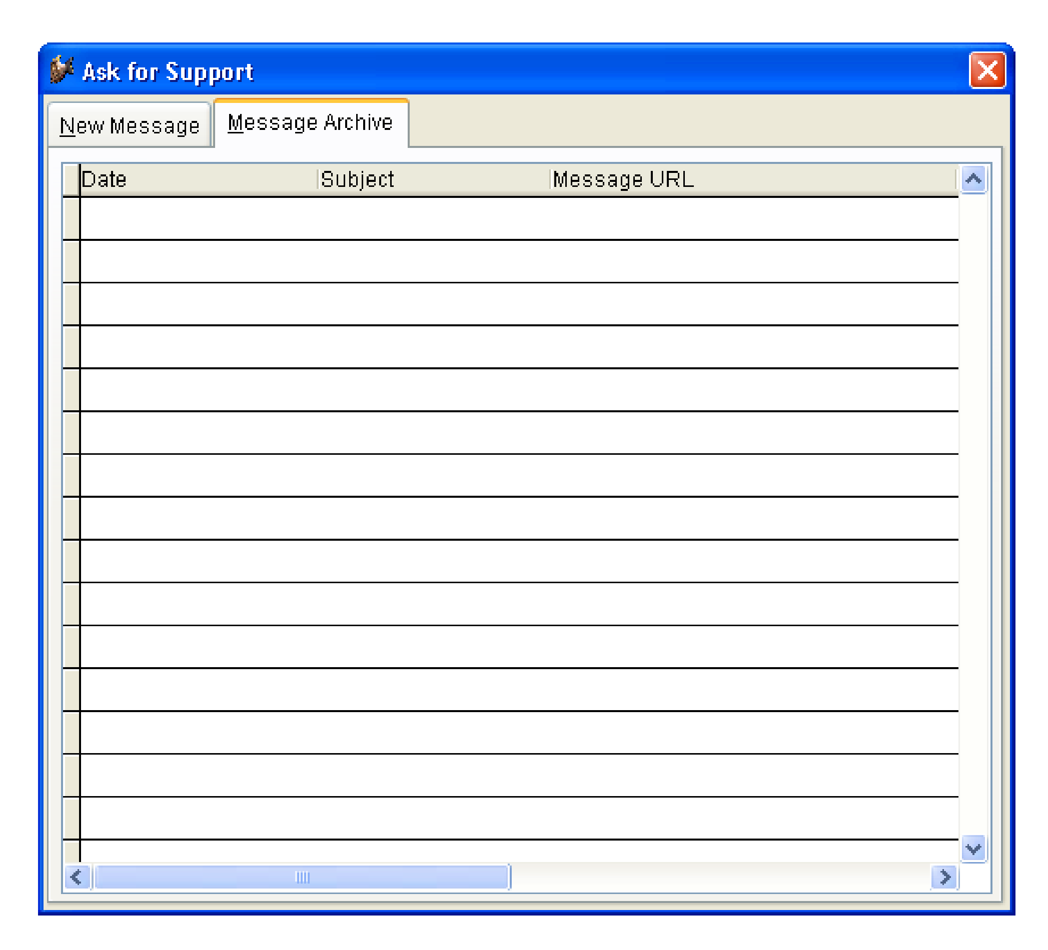
19.9.4. So erreichen Sie uns

Über den Menüpunkt *VFX Help, How to reach us* wird Ihnen angezeigt, wie Sie uns erreichen können.

19.9.5. Support-Anfragen an das Forum richten

Es gibt zwei Möglichkeiten um direkt aus VFX eine Support-Anfrage im Forum zu publizieren. Zum einen kann aus dem VFX-Menü über den Menüpunkt *VFX Help, Visual Extend Online, Forum* der Internet Browser gestartet werden. Es öffnet sich die Startseite des Forums. Hier können Nachrichten online gelesen und auch neue Nachrichten erstellt werden. Zum anderen kann über den Menüpunkt *VFX Help, Ask for Support* ein Dialog geöffnet werden, in dem offline eine Forumsnachricht verfasst werden kann. Zum Versenden einer Nachricht ist eine Internetverbindung erforderlich. Die auf diesem Weg verfassten Nachrichten bleiben gespeichert und können später auf der Seite *Message Archive* angesehen werden.


Ask for Support
New Message
Message Archive
My e-mail
Uwe.Habermann@dFPUG.de
Subject
Section
English
German
French
OK
Cancel



19.9.6. Info

Informationen über die installierte Visual Extend Version sowie Registrierungsinformationen erhalten Sie über den Menüpunkt *VFX Help, About Visual Extend*.

19.10. Neue Eigenschaften des Anwendungsobjekts

lHideRegistrationFiles – Wenn der Wert dieser Eigenschaft mit *.T.* eingestellt wird, werden die Dateien *VFX.ini* und *FirstInstall.txt* mit dem Attribut *hidden* gekennzeichnet. *.T.* ist der Standardwert.

lInstallClickyes – Mit dieser Eigenschaft kann eingestellt werden, ob die Anwendung *ClickYes* automatisch installiert werden soll, wenn E-Mails mit Outlook versendet werden sollen. Die Anwendung *ClickYes* unterdrückt die Sicherheitsfragen von Outlook. Der Standardwert dieser Eigenschaft ist *.T.*

lTableprompt – Wenn der Wert dieser Eigenschaft auf .T. gestellt wird, gilt die VFP-Einstellung *SET TABLEPROMPT ON*. Wenn ein Befehl ausgeführt wird, der eine geöffnete Tabelle oder Ansicht voraussetzt, erscheint bei Bedarf automatisch ein Datei-Öffnen-Dialog. Wenn der Wert dieser Eigenschaft auf .T. und es ist keine Tabelle oder Ansicht im aktuellen Arbeitsbereich geöffnet, kommt es zu einem Laufzeitfehler.

lVarcharmapping – Mit dieser Eigenschaft kann eingestellt werden, ob Zeichentypen mit variabler Feldlänge in Zeichentypen mit fester Feldlänge konvertiert werden sollen. Wenn der Wert *.T.* eingestellt wird, werden Datentypen mit variabler Zeichenfeldlänge von remote Datenquellen im VFP Feldtyp *Varchar* abgebildet. Der Standardwert ist *.F.* Dies entspricht der VFP Einstellung *SET VARCHARMAPPING OFF*.

nAllowSaveEmptyRecords – Verhindert das Speichern von leeren Datensätzen in allen Formularen, wenn der Wert *2* eingestellt wird. Wenn der Wert *1* eingestellt wird, können Benutzer leere Datensätze speichern. Beim Wert *0* kann individuell je Formular mit der Eigenschaft *lAllowSaveEmptyRecords* eingestellt werden, ob leere Datensätze gespeichert werden dürfen. Der Standardwert ist *0*.

nComboBoxListEntriesLimit – Der Wert dieser Eigenschaft gibt die Anzahl der Einträge einer Liste in einer aufgeklappten Combobox an. Der Standardwert ist *15*.

nReprocess – Dieser Wert gibt die Anzahl der Wiederholungsversuche für erfolglose Satz- oder Dateisperren an. Der Standardwert ist *0*.

nUseReportbehavior80forPDFOutput – Wenn Berichtsausgaben in eine PDF-Datei mit der Einstellung *SET REPORTBEHAVIOR 90* erfolgen, ist es nicht möglich aus der PDF-Datei Texte zu kopieren. Mit dieser Eigenschaft wird es ermöglicht PDF-Ausgaben mit der Einstellung *SET REPORTBEHAVIOR 80* zu erstellen, sodass Anwender Text kopieren können. Wenn der Wert dieser Eigenschaft *0* ist, kann je Formular mit der Eigenschaft *lUseReportbehavior80forPDFOutput* das Verhalten eingestellt werden. Wenn der Wert *1* ist, wird in allen Formularen *SET REPORTBEHAVIOR 80* bei der Erstellung von PDF-Ausgaben verwendet. Wenn der Wert *2* ist, wird die aktuelle Einstellung von *SET REPORTBEHAVIOR* verwendet.

19.11. Weitere Verbesserungen für Entwickler

Das Formular für die Aktivierung von VFX ist jetzt lokalisiert.

Im VFX – Messagebox Builder und im VFX – Message Editor können jetzt die Texte aller Sprachen bearbeitet werden, unabhängig von den Unicode-Einstellungen des Betriebssystems.

In den VFX – Form Buildern steht jetzt eine größere Auswahl von VFX-Klassen zur Verfügung.

Die VFX – Form Builder verwenden als Standardsteuerelement für Felder vom Typ *Date* die Klasse *cPickDate*. Dadurch haben Anwender die Möglichkeit ein Datum aus einem Kalender auszuwählen.

Im VFX – Class Switcher kann jetzt aus einem Auswahldialog eine beliebige Klassenbibliothek und Klasse aus dem aktuellen Projekt gewählt. Die Anzeige der Klassen und Klassenbibliothek ist alphabetisch. Wenn versucht wird eine nicht geeignete Klasse zuzuweisen, erscheint ein Warnhinweis und der Vorgang wird nicht fortgesetzt.

Im VFX-Menü gibt es unter *Project* jetzt eine Möglichkeit vom aktuellen Projekt eine Archivdatei anzulegen. Dateiname ist der Projektname gefolgt von einem Zeitstempel.

Das Debug-Menü für die Entwicklungsumgebung kann jetzt im VFX – Application Builder eingeschaltet werden. Wahlweise kann manuell die Eigenschaft *lDebugMode* der Klasse *cFoxAppl* in der Klassenbibliothek *Appl.vcx* auf .T. eingestellt werden.

In *Vfxfunc.prg* wurde die neue Funktion *GetNewGUID* hinzugefügt, die einen global und für immer eindeutigen ID-Wert zurückgibt. Die GUID wird mithilfe der API-Funktion *CoCreateGuid* ermittelt. Die Länge eines GUID ist 36 Zeichen. GUIDs können insbesondere dann als ID in Tabellen verwendet werden, wenn ein Datenabgleich mit anderen Tabellen vorgesehen ist.

Bei der Aktualisierung von Child-Daten in OneToMany-Formularen werden jetzt auch die Felder mit dem Benutzernamen (*ins_usr, edt_usr*) und dem Timestamp (*ins_date, edt_date*) von VFX automatisch mit Daten gefüllt.

In der Klasse *cFormbase* gibt es im Ereignis *Destroy* einen neuen Hook. Damit wird eine Eingriffsmöglichkeit für eigenen Code beim Schließen eines Formulars gegeben.

20. Neue Eigenschaften für Endbenutzer

20.1. Erforderliche Rechte zur Ausführung

VFX Anwendungen können für Windows XP zertifiziert werden. Zur Ausführung einer VFX Anwendung sind Windows-Standard-Benutzerrechte ausreichend. Entsprechend den Windows-Design-Richtlinien, können die ausführbaren Programmdateien (Exe-Datei, VFX.fll) unter *C:\Programme* von einem Installationsprogramm installiert werden. Alle anderen verwendeten Dateien können in anderen Ordnern installiert werden. Standardmäßig werden alle von einer Anwendung selbst erstellten Dateien im Ordner *C:\Dokumente und Einstellungen\All Users\Anwendungsdaten\<Firmenname>\<Anwendungsname>* gespeichert.

Zum Finden der Dateien Vfxpath.dbf oder Config.vfx wird die folgende Suchstrategie verwendet:

- Installationsordner der Anwendung (Exe-Datei) (aus Kompatibilitätsgründen zu früheren Versionen)
- *C:\Dokumente und Einstellungen\All Users\Anwendungsdaten\<Firmenname>\<Anwendungsname>*
- *C:\Dokumente und Einstellungen\<Aktueller Windows-Anmelde-name>\Anwendungsdaten\<Firmenname>\<Anwendungsname>*

Wenn eine dieser Dateien neu erstellt werden muss, wird zunächst versucht diese Datei im Ordner *C:\Dokumente und Einstellungen\All Users\Anwendungsdaten\<Firmenname>\<Anwendungsname>* zu speichern. Wenn der aktuelle Benutzer in diesem Ordner keine Schreibrechte hat, wird die Datei im Ordner *C:\Dokumente und Einstellungen\<Aktueller Windows-Anmeldename>\Anwendungsdaten\<Firmenname>\<Anwendungsname>* gespeichert.

Wenn mit einer remote Datenbank gearbeitet wird, wird die von VFP verwendete freie Tabelle zur Speicherung von *Autocomplete*-Werten *VFXAComp.dbf* nach der gleichen Strategie wie oben beschrieben gesucht und gespeichert.

20.2. Neue Icons

Viele neue Icons wurden für Endbenutzer erstellt und geben den Anwendungen ein deutlich verbessertes Erscheinungsbild. Auch in die Builder von VFX wurden zahlreiche neue Icons integriert und verbessern die intuitive Bedienung für Entwickler.

20.3. Datenzugriff

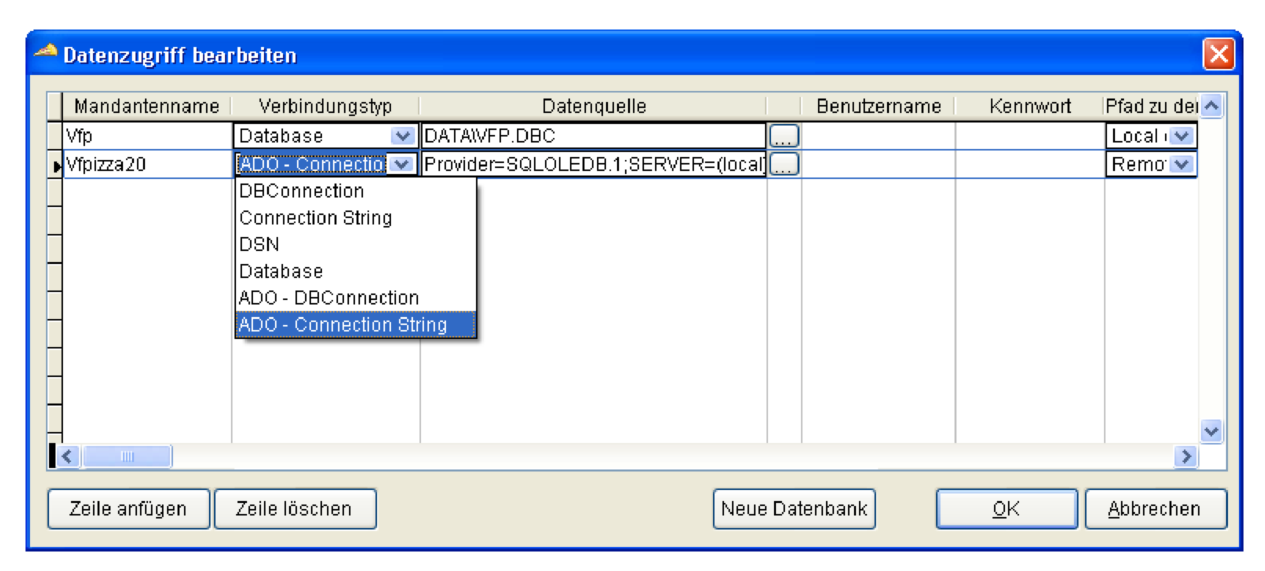
20.3.1. Der Dialog Datenzugriff bearbeiten

Zusätzlich zu den VFX 9.0 bekannten Möglichkeiten des Datenzugriffs kann in VFX 11.0 ein OLE-DB Provider zum Zugriff auf die Daten verwendet werden. Die Funktionalität der Klasse *cConnectionMgr* wurde erweitert um OLE-DB Verbindungen zu ermöglichen.

Im Dialog *Manage Config.vfx* kann zwischen drei OLE-DB Verbindungsmöglichkeiten gewählt werden:

ADO – DBConnection: Das ist eine Verbindung, die in einem Datenbank-Container gespeichert ist.
ADO – Connection String: Eine Verbindungszeichenfolge für einen OLE-DB Provider.

Diese beiden OLE-DB Verbindungstypen entsprechen etwa den ODBC-Verbindungstypen DBConnection und Connection String.



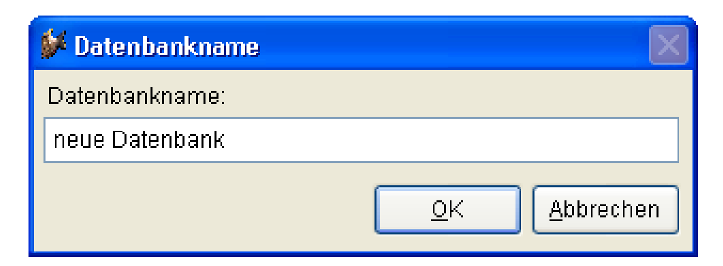
Die VFX-Tabellen können jetzt wahlweise auch in einer eigenen Remote Datenbank gespeichert werden und stehen so auf Wunsch Datenbank-Übergreifend zur Verfügung. Alle Verbindungstypen stehen auch für die Datenbank mit den VFX-Tabellen zur Verfügung.

Anlegen einer neuen Datenbank beim Kunden

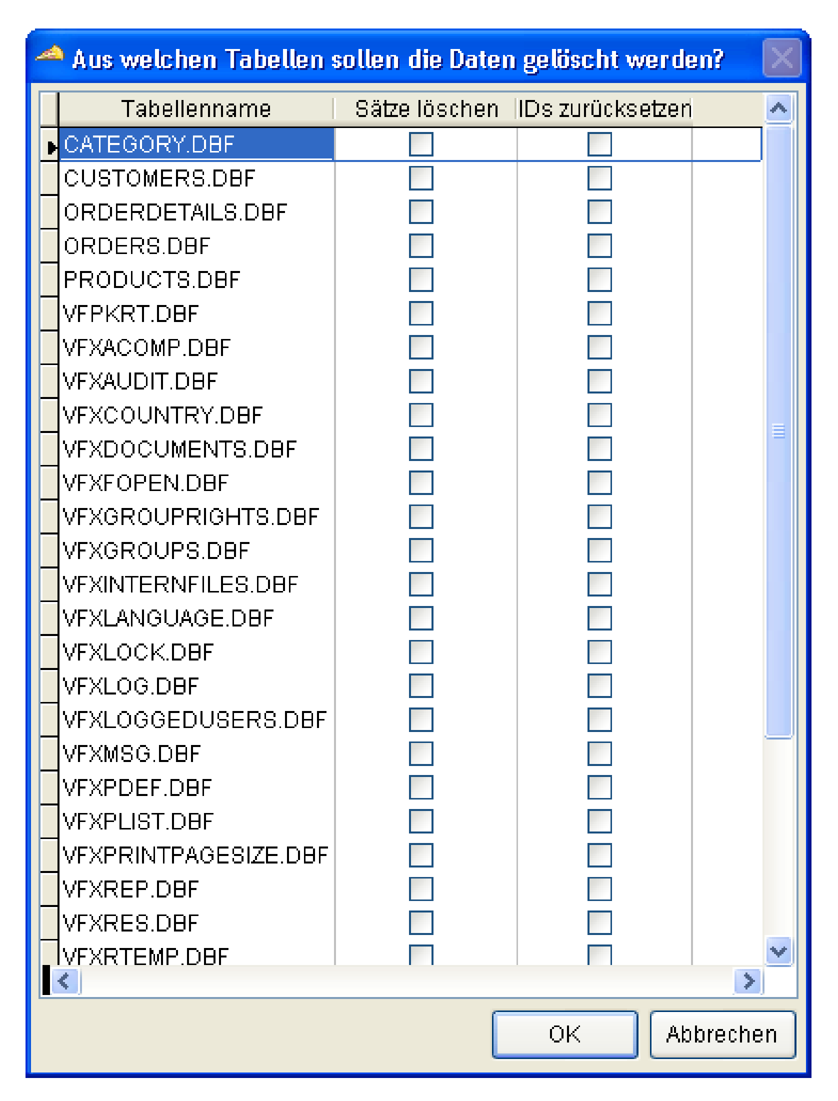
In VFX Anwendungen kann auf einfachem Weg eine neue Datenbank angelegt werden. Zur Laufzeit ist im Dialog *Datenzugriff bearbeiten* die Schaltfläche *Neue Datenbank* sichtbar. Die neue Datenbank wird mit der Struktur der aktuell selektierten Datenbank angelegt.

Wenn die aktuelle Datenbank eine VFP-Datenbank ist, erscheint ein Ordnerauswahldialog und der Benutzer kann einen neuen Ordner für die neue Datenbank anlegen.

Wenn die aktuelle Datenbank eine Remote Datenbank ist, kann der Benutzer der neuen Datenbank in einem Dialog einen Namen geben.



Im nächsten Dialog wird gefragt, aus welchen Tabellen die Daten in der neu angelegten Datenbank gelöscht werden sollen. Auf Wunsch können auch die ID-Werte für einzelne Tabellen zurückgesetzt werden.



20.4. Neue Eigenschaften in Onetomany-Formularen

Die Daten aus Child-Grids auf OneToMany-Formularen können jetzt per Drag & Drop in andere Anwendungen kopiert werden. Diese Option kann im VFX – COneToMany Builder und im VFX – CChildgrid Builder ein- bzw. ausgeschaltet werden.

In VFX – OneToMany Builder und im VFX – CchildGrid Builder kann für jede Spalte eines Child Grid eingestellt werden, ob diese Spalte beim OLE Drag & Drop mit kopiert wird.

Zu jeder Spalte in einem Child-Grid auf einem OneToMany-Formular kann eine Summe gebildet werden. Diese Option kann im VFX – COneToMany Builder und im VFX – CChildgrid Builder ein- bzw. ausgeschaltet werden. Wenn eine Summe gebildet werden soll, wird am unteren Formularrand ein Label mit der Bezeichnung der Spalte sowie eine Textbox mit der Summe hinzugefügt.

Das Datum sowie ggf. die Zeit und der Benutzername werden jetzt auch bei Child-Datensätze protokolliert, wenn die entsprechenden Felder in der Child-Tabelle vorhanden sind.

20.4.1. OnChildRequery

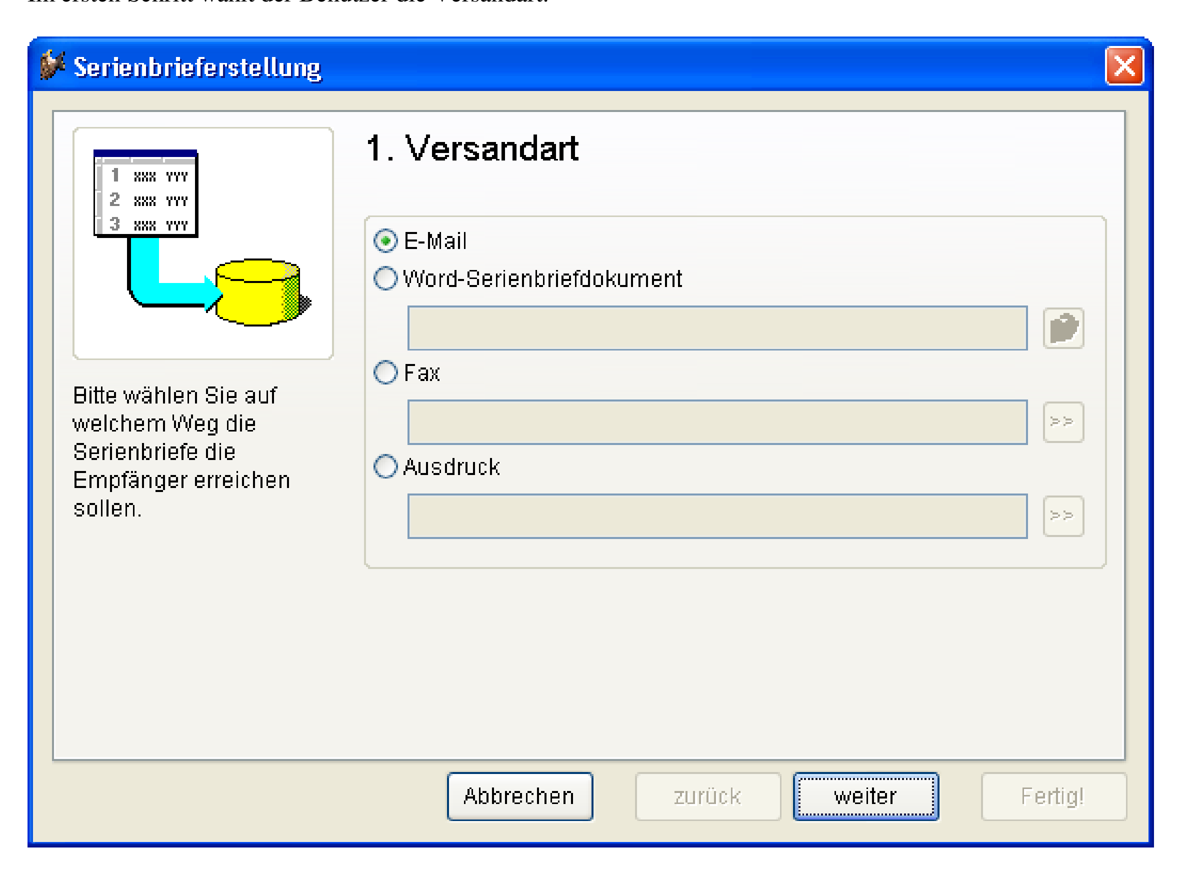
In früheren VFX-Versionen musste Code in die Methode *OnChildRequery* von OneToMany-Formularen eingetragen werden, wenn die Child-Daten auf Ansichten oder CursorAdaptern basierten.

Dies ist nicht mehr erforderlich. Beim Bewegen des Satzzeigers im Parent-Teil des Formulars werden automatisch alle Child-Arbeitsbereiche überprüft. Wenn ein Child-Arbeitsbereich auf einer Ansicht oder einem CursorAdapter basiert, werden die Daten aktualisiert. Bei Ansichten wird dazu *REFRESH()* aufgerufen. Bei CursorAdaptern wird die Methode *CursorRefresh()* ausgeführt.

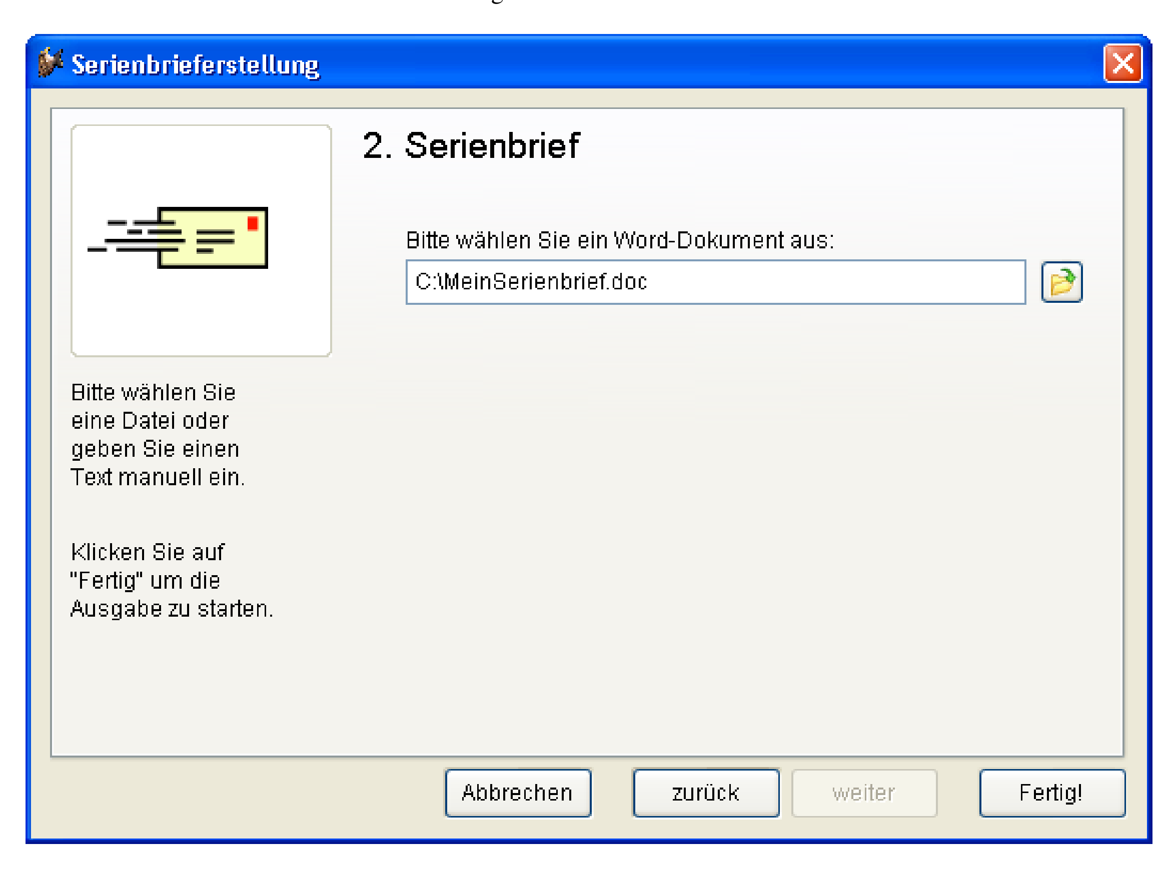
20.5. Seriendokumenterstellung

Mit dem Assistenten zur Seriendokumenterstellung kann dem Benutzer die Möglichkeit gegeben werden Seriendokumente basierend auf den Daten der Anwendung zu erstellen. Als Text für die generierten Dokumente kann ein Word-Serienbriefdokument oder eine Textdatei verwendet werden oder es kann manuell im Assistenten ein Text eingegeben werden. Das Ergebnis der Seriendokumentausgabe kann wahlweise als Word-Dokument gespeichert werden, gedruckt werden, als Fax gesendet werden oder als E-Mail gesendet werden. Der Benutzer wird durch den Assistenten in wenigen intuitiven Schritten geführt.

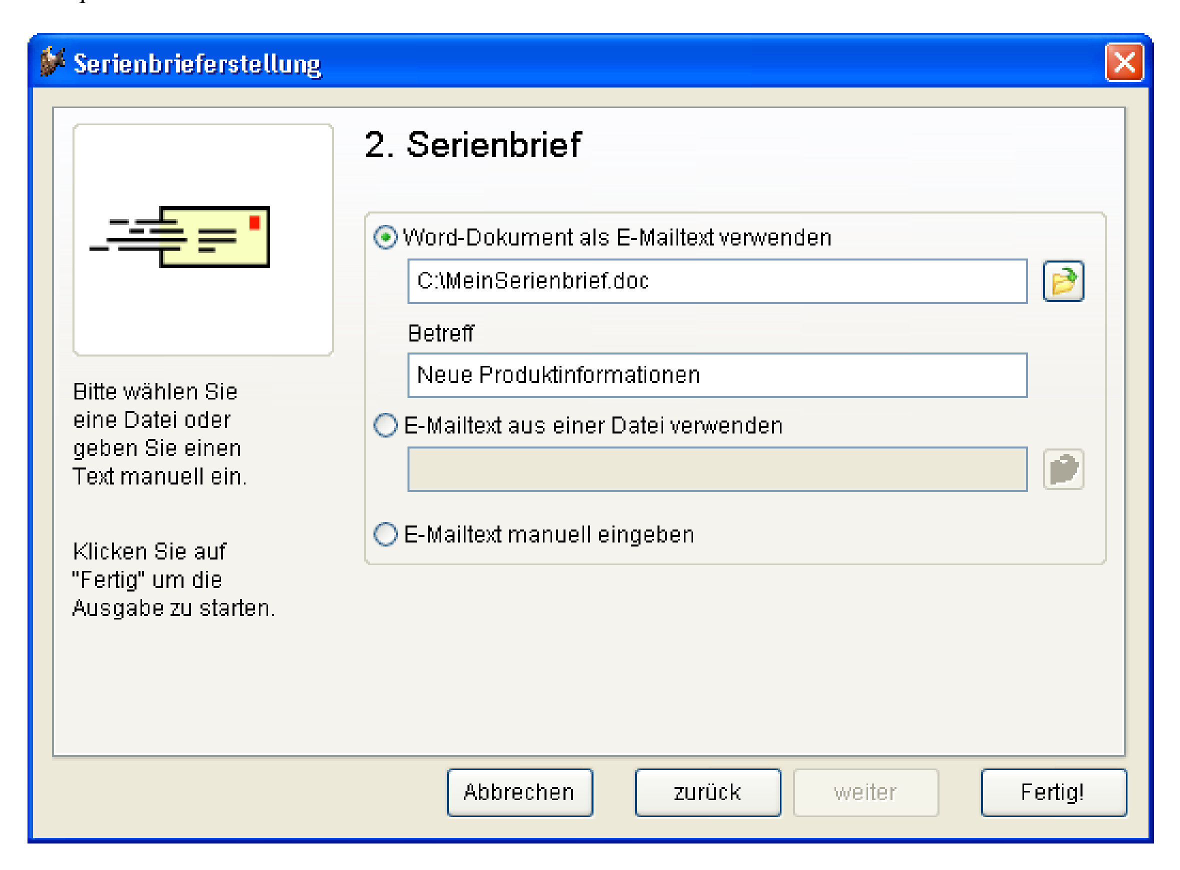
Im ersten Schritt wählt der Benutzer die Versandart.



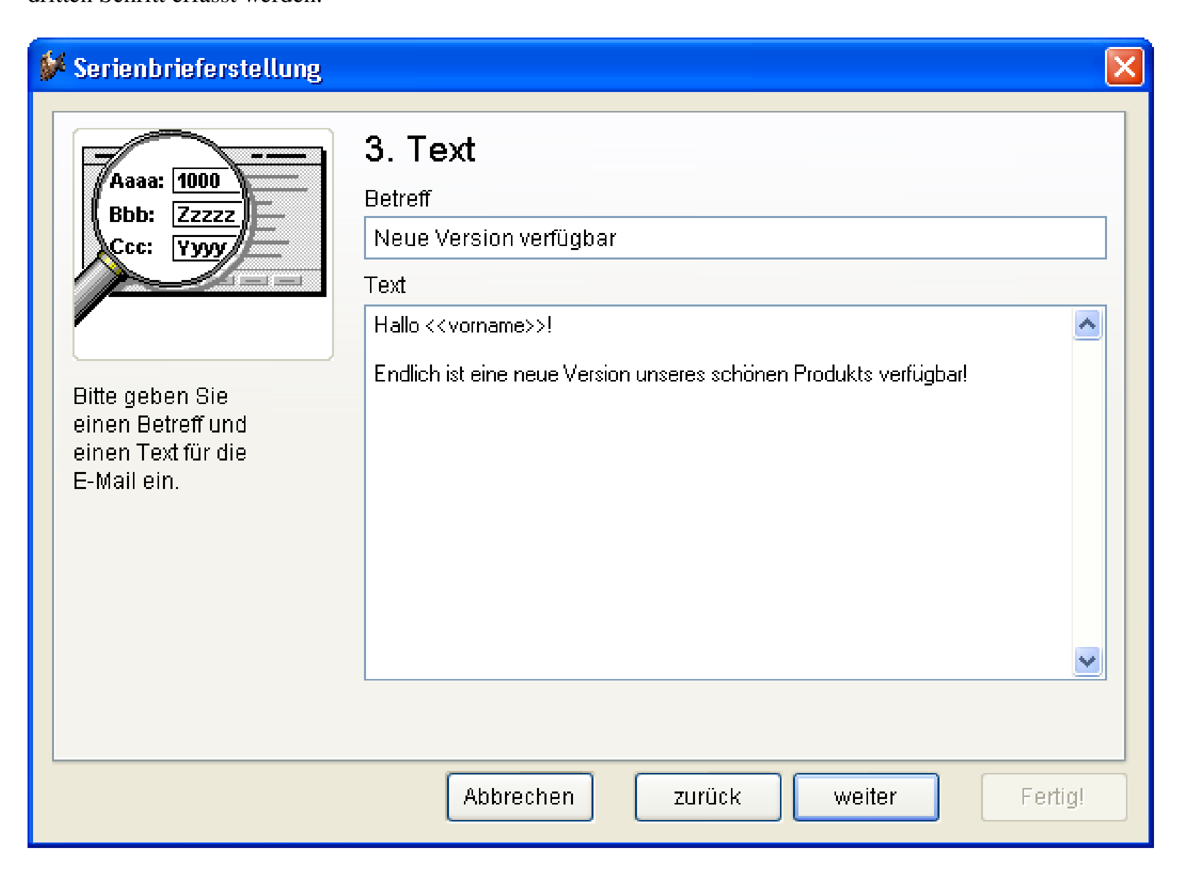
Im zweiten Schritt wird der Text für die zu erstellenden Dokumente ausgewählt. Wenn im ersten Schritt Word-Serienbriefdokument, Fax oder Ausdruck gewählt wurde, kann der Benutzer in disem Schritt den Daei- und Pfadnamen des zu erstellenden Dokuments eingeben.



Wenn im ersten Schritt E-Mail ausgewählt wurde, kann der Benutzer im zweiten Schritt zwischen drei möglichen Textquellen wählen.



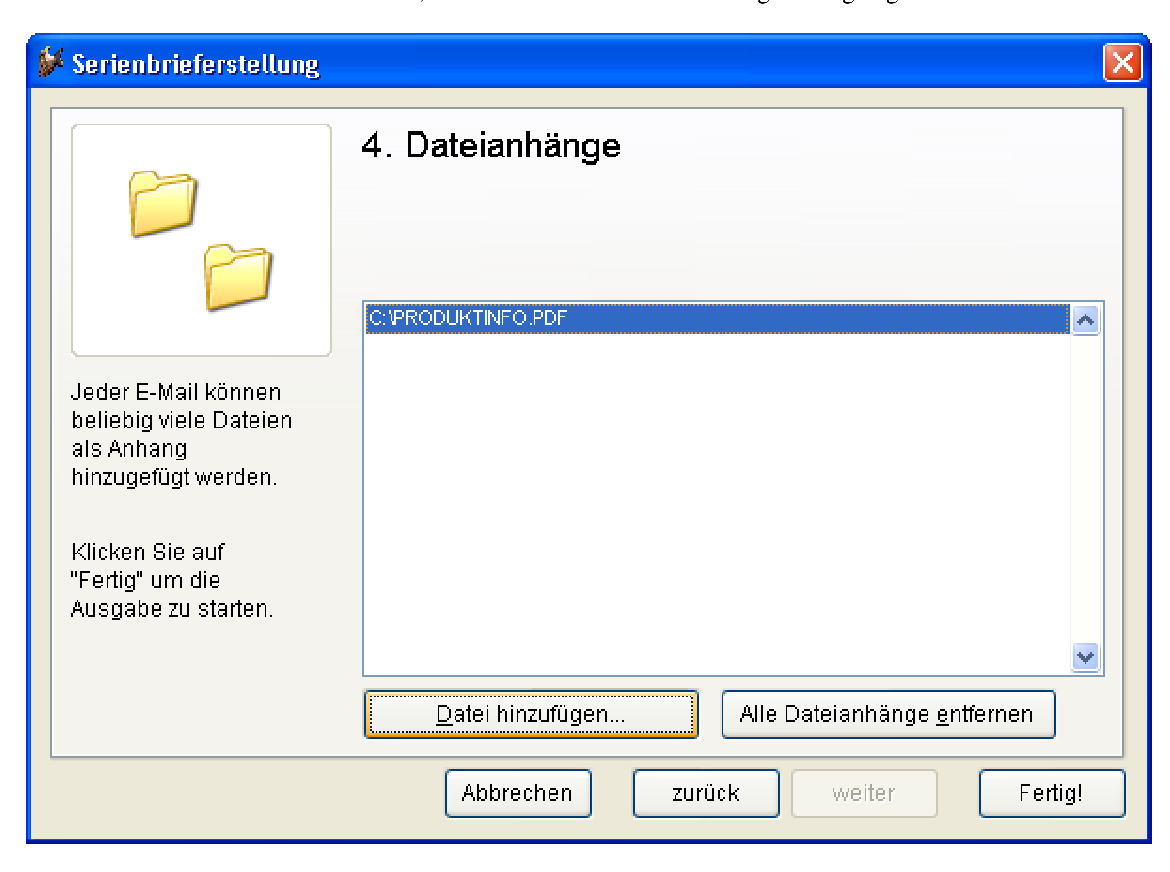
Wenn im zweiten Schritt ausgewählt wurde, dass ein Text manuell eingegeben werden soll, kann der Text im dritten Schritt erfasst werden.



Wenn im zweiten Schritt ein Textdokument ausgewählt wurde, kann dieser Text im dritten Schritt bei Bedarf geändert werden.

In Serienbriefdokumenten kann, ähnlich wie in Word, ein variabler Text verwendet werden. Dieser variable Text muss in speziellen Zeichen eingeschlossen sein. Standardmäßig sind diese Zeichen doppelte spitze Klammern, also „<<“ und „<<“. Die Begrenzungszeichen können vom Entwickler in den Eigenschaften *cLeftDelim* und *cRightDelim* der Klasse *cMailMerge* eingestellt werden.

Wenn E-Mails versendet werden sollen, können im vierten Schritt Anhänge hinzugefügt werden.



Durch einen Mausklick auf die Schaltfläche *Fertig* werden die Seriendokumente erstellt. Im letzten Schritt wird dem Benutzer die Anzahl der erstellten Seriendokumente angezeigt.

20.5.1. Die Klasse cMailMerge

Diese Formularklasse ist in der Klassenbibliothek *Vfxform.vcx* gespeichert.

Mit dieser Klasse können Endanwender anspruchsvolle Seriendokumente erstellen. Folgende Optionen stehen zur Verfügung:

1. E-Mail
   Erstellen von Serien-E-Mails. Der E-Mailtext kann aus einem Word-Dokument oder einer Textdatei stammen oder auch manuell in einer Editbox eingegeben werden. Wenn eine Text-E-Mail erstellt wird, können zusätzlich beliebig viele Dateien als Anhang versendet werden.
2. Word-Dokument
   Erstellen einer Word-Serienbriefausgabe basierend auf einem Word-Serienbriefdokument. Die Word-Serienbriefausgabe kann dann in Word beliebig weiterbearbeitet werden.
3. Fax
   Versenden von Serienfaxen basierend auf einem Word-Serienbriefdokument.
4. Drucken
   Drucken von Serienbriefen basierend auf einem Word-Serienbriefdokument.

Zur Seriendokumenterstellung muss ein Cursor vorhanden sein, der die erforderlichen Felder für jede mögliche Benutzerauswahl enthält.

Eigenschaften

cDataSource	Enthält den Namen der Datenquelle für die Seriendokumenterstellung. Diese Datenquelle wird von Word oder vom Assistenten direkt verwendet. Alle variablen Felder müssen in dieser Datenquelle enthalten sein.

cMailAddressFieldName Enthält den Namen des Feldes für die E-Mailadresse. Dieser Feldname muss in dem in *cDataSource* angegebenen Cursor enthalten sein. Diese Eigenschaft wird nur beim Versand von E-Mails verwendet.

cCCFieldName Enthält den Feldnamen für eine CC E-Mailadresse. Dieser Feldname muss in dem in *cDataSource* angegebenen Cursor enthalten sein. Diese Eigenschaft wird nur beim Versand von E-Mails verwendet.

cBCCFieldName Enthält den Feldnamen für eine BCC E-Mailadresse. Dieser Feldname muss in dem in *cDataSource* angegebenen Cursor enthalten sein. Diese Eigenschaft wird nur beim Versand von E-Mails verwendet.

cFaxNumberFieldName Enthält den Namen des Feldes mit der Faxnummer. Dieser Feldname muss in dem in *cDataSource* angegebenen Cursor enthalten sein. Diese Eigenschaft wird nur beim Faxversand verwendet.

cLeftDelim Linkes Begrenzungszeichen für die Ersetzung von Text. Standardwert ist „<<“. Die hier angegebene Zeichenkette ist die linke Begrenzung eines variablen Wertes.

cRightDelim Rechtes Begrenzungszeichen für die Ersetzung von Text. Standardwert ist „>>“. Die hier angegebene Zeichenkette ist die rechte Begrenzung eines variablen Wertes.

cMergeText Nur zur internen Verwendung. Hier ist der Serientext gespeichert.

nEmailsSent Nur zur internen Verwendung. Zähler für die Anzahl der zu erstellenden Dokumente. Die Anzahl wird auf der letzten Seite des Wizard angezeigt.

nPreviousPageNum Nur zur internen Verwendung. Enthält die Nummer der vorhergehenden Seite im Wizard.

Methoden

LoadFileContent Lädt die im zweiten Schritt angegebene Textdatei. Der Text kann im dritten Schritt in einer Editbox bearbeitet werden.

SendMails Erstellen der Seriendokumente entsprechend der gewählten Optionen. Von hier wird eine der Methoden *SendThroughMapi* oder *SendThroughOleWord* aufgerufen.

SendThroughMapi Erstellt Serien-E-Mails unter Verwendung der VFX-Klasse *cEmail*.

SendThroughOleWord Diese Methode erstellt Serienbriefe per OLE-Automatisierung von Word. Auf diesem Weg kann die Serienbriefausgabe in ein Word-Dokument gespeichert, per Fax gesendet, gedruckt oder per E-Mail versendet werden.

20.6. Berichte

In VFX Anwendungen ist die Standardeinstellung für Berichtsausgaben *Set Reportbehavior 90*. Damit können alle neuen Eigenschaften des Berichts-Designers und der Berichts-Engine von VFP 9 genutzt werden. Mit der Eigenschaft *nReportBehavior* des Anwendungsobjekts kann auf Wunsch das Reportbehavior auf 80 eingestellt werden. Diese Eigenschaft kann auch im VFX – Application Builder eingestellt werden.

Alle Berichte, die auf Grids basieren und zur Laufzeit erzeugt werden, können jetzt mit mehrzeiligen Detailbändern ausgegeben werden. Mit der Eigenschaft *nMultiLineReport* des Anwendungsobjekts kann das Verhalten eingestellt werden. Ein Wert von 0 bedeutet, dass mit der Eigenschaft *lMultiLineReport* auf Formularebene mehrzeilige Berichte aktiviert werden können. Bei einem Wert von 1 werden mehrzeilige

Berichte in allen Formularen aktiviert. Mit dem Wert 2 wird das Verhalten von früheren VFX-Versionen eingestellt. Berichte werden einzeilig, bis zum rechten Papierrand bedruckt.

20.6.1. Berichte bearbeiten

Endbenutzer haben die Möglichkeit Berichtsdateien selbst zu bearbeiten. Dafür muss sich die Datei *Vfxmodifyreport.exe* im gleichen Ordner wie die Anwendung befinden. Außerdem muss der aktuelle Benutzer das Recht zur Berichtsbearbeitung haben. Dieses Recht kann in der Benutzerverwaltung sowie für Benutzergruppen für jeden Benutzer individuell von Administratoren mit der Benutzerstufe 1 eingestellt werden.

Der Start der Anwendung *Vfxmodifyreport.exe* aus dem Windows-Explorer ist nicht möglich. Eine unberechtigte Benutzung dieser Anwendung wird so verhindert.

Die Anwendung zur Bearbeitung der Berichtsdateien befindet sich aus Sicherheitsgründen in einer eigenen ausführbaren Datei. Diese Datei erhält beim Start als Parameter die aktuell eingestellte Sprache der Anwendung und startet somit lokalisiert. Zu beachten ist, dass einige der Dialoge von der Laufzeitumgebung von VFP stammen und damit in jedem Fall in der Sprache der VFP Laufzeitumgebung angezeigt werden.

Entwickler, die diese Anwendung programmatisch starten wollen, können als zweiten Parameter den Namen einer Berichtsdatei übergeben. Beim Start aus dem Menü *Extras* der Anwendung erscheint ein Öffnen-Dialog zum Öffnen einer Berichtsdatei.

Berichte werden in jedem Fall mit der Klausel *PROTECTED* des *MODIFY REPORT* Befehls bearbeitet. Dadurch können alle Schutzeinstellungen von Reportbehavior 90 genutzt werden.

20.6.2. ReportOutput und ReportPreview

Der Quellcode der VFP-Berichtsanwendungen *ReportOutput.app* und *ReportPreview.app* ist in VFX integriert worden. Selbstverständlich wurde der Code so angepasst, dass sowohl die Lokalisierung zur Entwicklungszeit als auch zur Laufzeit unterstützt werden.

Die Anwendung ReportOutput enthält alle verfügbaren ReportListener. Zusätzlich gibt es einen neuen ReportListener zur Erstellung von PDF-Dateien.

In der Seitenansicht von Berichten steht ein Rechtsklickmenü zur Verfügung. Hierüber kann der Benutzer die Berichtsausgabe drucken, in eine Datei speichern oder als E-Mail versenden.

Für das Drucken von Berichten steht ein erweiterter Druckdialog zur Verfügung. In diesem Dialog können die zu druckenden Seiten gewählt werden. Es ist auch möglich mehrere Seiten eines Berichts auf einer Seite zu drucken. Die Sortierfolge der Seiten kann eingestellt werden.

20.6.3. PDF-ReportListener

Dieser neue ReportListener wurde zusätzlich zu den bereits in VFX 9.0 vorhandenen ReportListenern hinzugefügt. Mit diesem ReportListener ist die Berichtsausgabe in PDF-Dateien möglich. Die Installation von Ghostscript ist nicht erforderlich.

Der PDF-ReportListener verwendet Dateien mit einer beträchtlichen Größe. Diese Dateien sind daher nicht in VFX-Anwendungen eingeschlossen. Die benötigten Dateien befinden sich im Projekt *PDFOutput.pjx*. Die aus diesem Projekt erstellte App-Datei kann im Exe-Ordner der Anwendung installiert werden und wird dann automatisch verwendet. Wahlweise kann die Datei *PDFOutput.app* in einem Zip-Archiv bei Bedarf automatisch aus dem Internet heruntergeladen werden. Der Download-Link befindet sich in der Tabelle *Vfxsys.dbf* im Feld *Install_GS*. Standardmäßig wird diese Datei von der Visual Extend-Webseite heruntergeladen.

```
D: http://files.visualextend.com/files95/PDFOutput.Zip
```

Hinweis: In früheren Versionen von VFX befand sich im Feld *Vfxsys.Install_GS* das Installationsskript für GhostScript. Wenn die Download Option von *PDFOutput.app* genutzt werden soll, ist in dieses Feld manuell der Download-Link einzutragen.

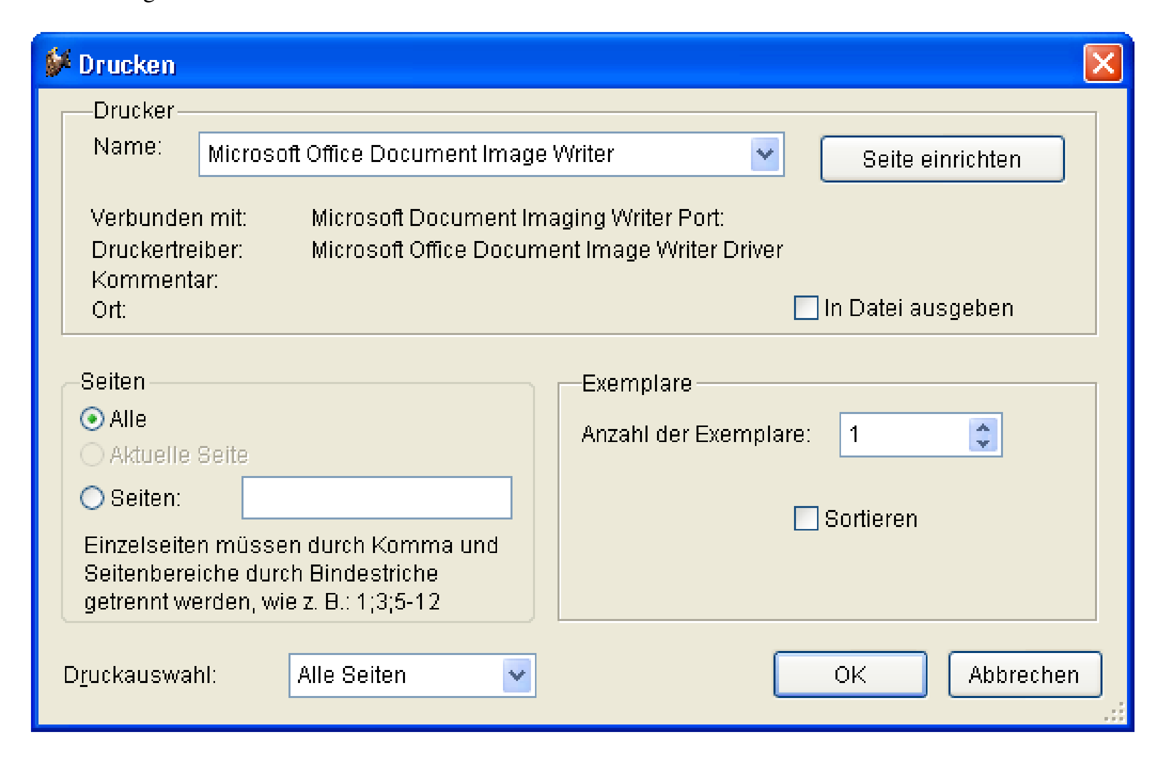
Die Installation von GhostScript ist nicht erforderlich.

20.6.4. Erweiterter Druckdialog

In VFX Anwendungen kann ein erweiterter Druckdialog verwendet werden, der es den Benutzern erlaubt die Druckausgaben genauer einzustellen. Der neue Druckdialog basiert auf der Klasse *cPrintDialog* und benutzt die neue Klasse *cPrintEngine*.

20.6.5. Die Klasse cPrintDialog

Die Klasse *cPrintDialog* ist in der Klassenbibliothek *Vfxform.vcx* gespeichert. Diese Klasse zeigt den erweiterten Druckdialog an.



In diesem Dialog kann der Drucker ausgewählt werden, die Seiteneinstellungen können verändert werden, die Ausgabe kann in eine Datei umgelenkt werden, die Anzahl der Exemplare kann eingestellt werden und die zu druckenden Seiten können ausgewählt oder eingegeben werden.

Eigenschaften

cPageRange – In dieser Eigenschaft steht die Auswahl der zu druckenden Seiten. Dieser Wert wird nur berücksichtigt, wenn ein manuell eingegebener Seitenbereich gedruckt werden soll (*nPagesSelectionType* = 3).

nAllOddEven - Auswahl zu druckender Seiten.
1 – Alle
2 – Ungerade Seiten
3 – Gerade Seiten

nCollate – Sortierfolge der Seiten.
0 – Exemplare werden nacheinander gedruckt.
1 – Zu allen Exemplaren wird zunächst die erste Seite gedruckt, dann werden alle zweiten Seiten gedruckt usw.

nNumberOfCopies – Anzahl zu druckender Exemplare.

nPagesSelectionType - Seitenauswahl
1 – Alle Seiten
2 – Aktuelle Seite
3 – Seitenbereich

nPrintToFile – Numerischer Wert mit dem Ausgabeziel.
0 – Drucken

1 – Ausgabe in eine Datei

oUnderlyingObject – Referenz auf ein Objekt der Klasse *cPrintEngine*.

Printers – Array mit den Informationen aller installierten Druckertreiber.

20.6.6. Die Klasse cPrintEngine

Diese Klasse dient zum drucken von Berichten. Als Parameter wird eine Referenz auf ein ReportListener-Objekt übergeben. Diese Klasse hat Methoden um einen Druckauftrag zu starten, mit Seiten zu füllen und zu schließen. Abhängig vom Wert der Eigenschaft *goProgram.nCustomPrintDialog* kann ein erweiterter Druckdialog angezeigt werden.

Dieser Dialog ermöglicht den Drucker, die Druckereigenschaften, die Anzahl zu druckender Exemplare, eine Auswahl von Seiten und die Sortierung der Seiten zu wählen. Der erweiterte Druckdialog ist eine Ableitung von der Klasse *cPrintDialog*.

Eigenschaften

aPagesToPrint – In diesem Array werden die zum Drucken ausgewählten Seiten gespeichert. Das erste Element jeder Zeile enthält die Nummer der Startseite. Das zweite Element jeder Zeile enthält die Nummer der Endseite. Eine einzeln zu druckende Seite wird in beide Elemente einer Zeile eingetragen. Wenn alle Seiten gedruckt werden sollen, enthält dieses Array nur eine Zeile.

cPageRange - In dieser Eigenschaft steht die Auswahl der zu druckenden Seiten. Dieser Wert wird nur berücksichtigt, wenn ein manuell eingegebener Seitenbereich gedruckt werden soll (*nPagesSelectionType* = 3).

cPrinterDriver - Name des aktuell ausgewählten Druckertreibers.

cPrinterName - Name des aktuell ausgewählten Druckers.

cPrintFileName - Name der Datei, wenn die Ausgabe in eine Datei umgelenkt werden soll.

lCustomPrintDialog – Soll der erweiterte Druckdialog angezeigt werden?
.T. – Anzeige des erweiterten Druckdialogs.
.F. – Anzeige des Standarddruckdialogs.

nAOE – Auswahl der zu druckenden Seiten.
1 – Alle Seiten
2 – Nur ungerade Seiten
3 – Nur gerade Seiten

nHeight – Höhe des bedruckbaren Bereichs des aktuellen Druckers.

nLeft – Linker Rand des bedruckbaren Bereichs des aktuellen Druckers.

nPagesSelectionType – Seitenauswahl.
1 – Alle Seiten
2 – Aktuelle Seite
3 – Seitenbereich

nPrinterHandle – Handle des verwendeten Druckertreibers.

nPrintToFile – Numerischer Wert mit dem Ausgabeziel.
1 – Berichtsausgabe in eine Datei.
0 – Drucken.

nShowDialog – Nur zur internen Verwendung. Gibt an, ob der Druckdialog angezeigt wurde.

nTop – Oberer Rand des bedruckbaren Bereichs des aktuellen Druckers.

nWidth – Breite des bedruckbaren Bereichs des aktuellen Druckers.

oRepListener – Referenz zum aufrufenden ReportListener.

Methoden

EndPrinterJob()
Beendet einen Druckauftrag. Bei erfolgreicher Ausführung wird *.T.* zurückgegeben. Im Fehlerfall wird *.F.* zurückgegeben.

GetPageProp()
Ermittelt die Eigenschaften Breite, Höhe, oberer Rand und linker Rand des bedruckbaren Bereichs des ausgewählten Druckers und speichert die Werte in den Eigenschaften *nWidth, nHeight, nTop, nLeft*. Bei erfolgreicher Ausführung wird *.T.* zurückgegeben. Im Fehlerfall wird *.F.* zurückgegeben.

GetPagesToPrint()
Füllt das Array *aPagesToPrint* entsprechend den Werten der Eigenschaften *nPagesSelectionType* und *cPageRange*. Wenn keine gültigen Seiten ermittelt werden können, wird das Array verworfen und dem Benutzer wird eine Meldung angezeigt. Bei erfolgreicher Ausführung wird *.T.* zurückgegeben. Im Fehlerfall wird *.F.* zurückgegeben.

OpenPrintDialog()
Zeigt den erweiterten Druckdialog an. Wenn der Benutzer auf die Schaltfläche *OK* klickt, werden die Methoden *GetPagesToPrint()* und *StartPrinterJob()* aufgerufen. Bei erfolgreicher Ausführung wird *.T.* zurückgegeben. Im Fehlerfall wird *.F.* zurückgegeben.

Output_Page(nPageNo)
nPageNo – Gibt die Nummer der zu druckenden Seite an. Es wird die ReportListener-Methode *OutputPage()* verwendet um die Seite an den Druckertreiber zu senden.

Print_Pages()

Liest das Array *aPagesToPrint* und ruft für jede Seite die Methode *Output_Page(nPageNo)* auf. Abschließend wird die Methode *EndPrinterJob()* aufgerufen. Bei erfolgreicher Ausführung wird *.T.* zurückgegeben. Im Fehlerfall wird *.F.* zurückgegeben.

StartPrinterJob()

Stellt den im erweiterten Druckdialog selektierten Drucker als aktuellen Drucker ein und startet den Druckauftrag. Rückgabewert ist ein Handle zu dem Druckauftrag.

20.7. XP-Öffnen-Dialog

Die Funktionalität des XP-Öffnen Dialogs wurde so erweitert, dass darüber nicht nur Formulare geöffnet werden können, sondern wahlweise auch ein Befehl ausgeführt werden kann. Um Befehle auszuführen wird das Feld *Form* in der Tabelle *Vfxfopen* leer gelassen. Der auszuführende Befehl wird in das Feld *Parameter* eingetragen. Hierüber lassen sich insbesondere Prozeduren, Funktionen und Methoden aufrufen.

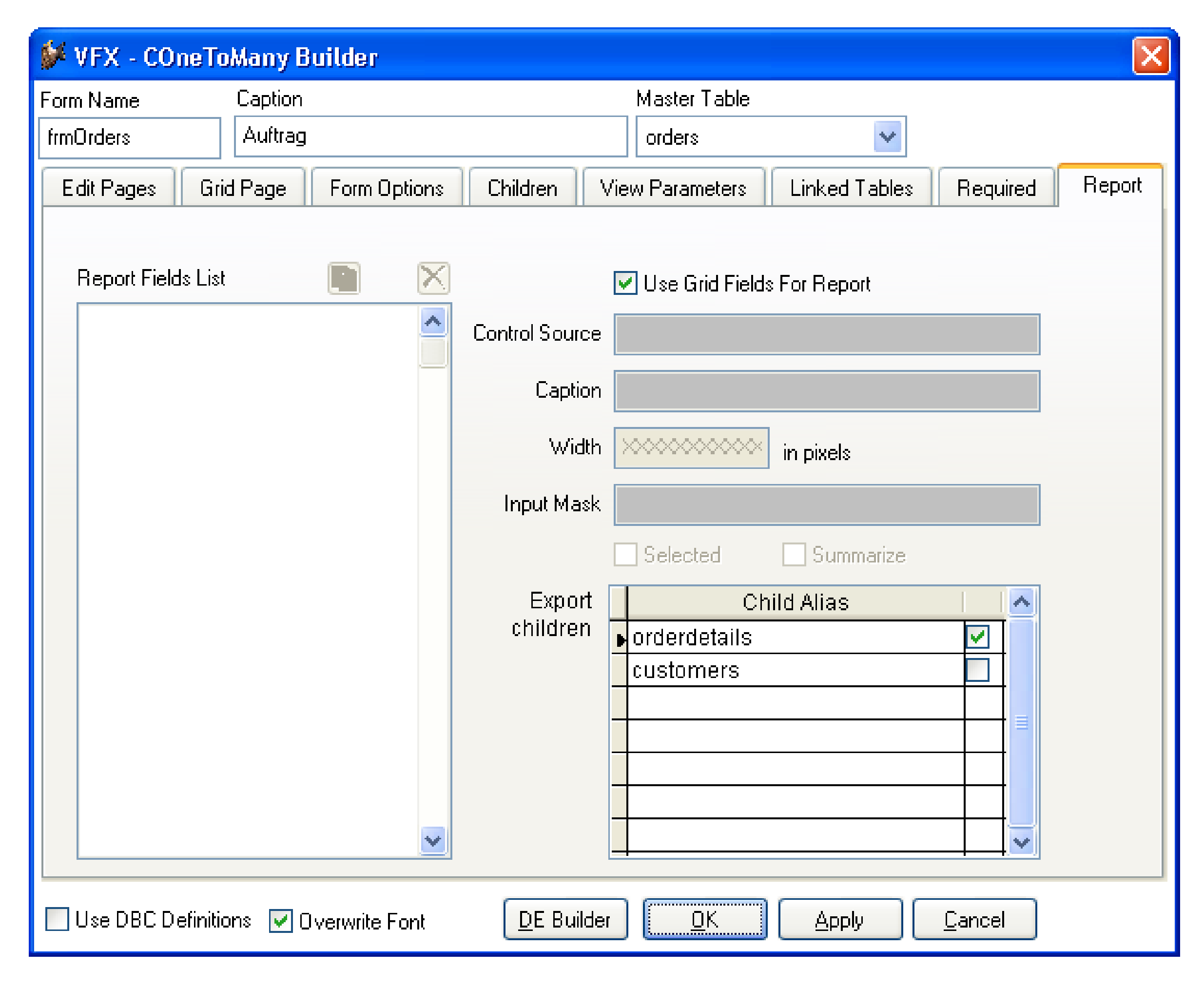
In der Tabelle *Vfxfopen.dbf* gibt es jetzt ein Feld *Iconfile*. In diesem Feld kann der Dateiname zu einem Icon zu dem aktuellen Formular gespeichert werden. Dieses Icon wird im XP-Öffnen-Dialog angezeigt.

Favoriten erscheinen jetzt auch im XP-Öffnen-Dialog in eigenen Gruppen je Formular. Jeder Benutzer kann im Anpassen-Dialog individuell für sich einstellen, ob Favoriten im XP-Öffnen-Dialog angezeigt werden sollen.

20.8. Datenexport

Im Menü für Endanwender gibt es im Menü unter *Datei* den neuen Menüpunkt *Export als*. Darunter gibt es die Auswahlmöglichkeiten *CSV, Excel, XML* und *DBF*. Diese Menüpunkte sind aktiviert, wenn ein Formular mit Daten geöffnet und aktiv ist. Die Auswahl einer dieser Optionen öffnet einen *Speichern unter*-Dialog. Nach Eingabe eines Dateinamens werden die Daten aus dem *Initialselectedalias* des Formulars in einer Datei mit dem gewählten Dateiformat gespeichert. Die aktuelle Sortierung sowie ein eventuell gesetzter Filter werden berücksichtigt. Es werden alle Felder exportiert.

Wenn als Exportformat *XML* gewählt wird, können aus Onetomany-Formularen wahlweise auch die Child-Daten exportiert werden. Diese Möglichkeit besteht nur dann, wenn die Parent- und Child-Daten über eine Relation in einer Beziehung stehen. Alle an der Relation beteiligten Child-Tabellen können im *XML*-Format exportiert werden. Im VFX – COneToMany Builder kann auf der Seite Report eingestellt werden, welche Child-Tabellen mit exportiert werden sollen.



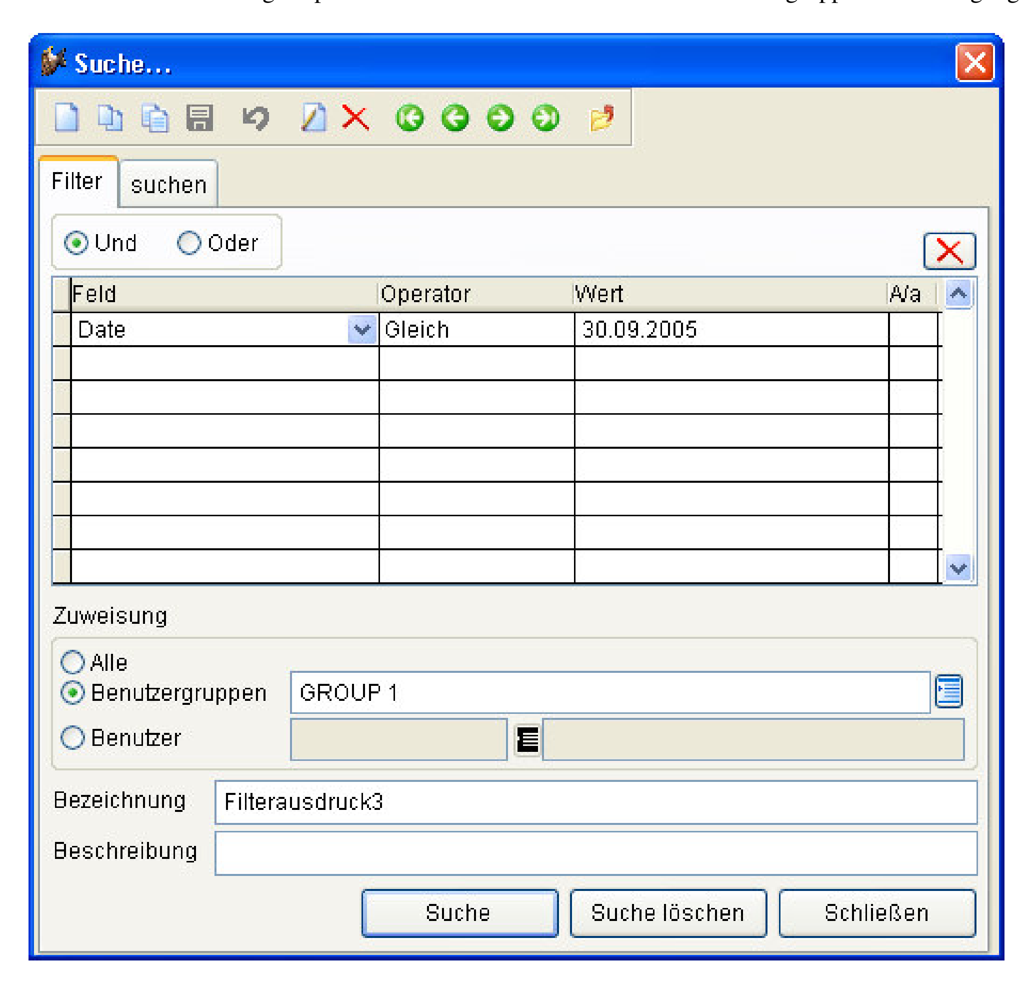
Wenn Child-Daten für den *XML*-Export markiert wurden, wird der Benutzer zur Laufzeit gefragt, ob die Child-Daten mit exportiert werden sollen.

20.9. Suchdialog

Filtereinstellungen können jetzt je Formular und je Benutzer oder Benutzergruppe gespeichert werden. Der Suchdialog wurde um Steuerelemente zur Verwaltung der Filterdefinitionen erweitert. Filterdefinitionen können für andere Benutzer kopiert werden.

Über die Eigenschaft *nFilterBehavior* in der Klasse *cFoxAppl* kann eingestellt werden, ob die neuen Eigenschaften des Suchdialogs genutzt werden sollen oder ob der Suchdialog die gleichen Funktionen wie in VFX 9.0 haben soll. Es ist auch möglich den gewünschten Suchdialog für jedes Formular einzustellen. Hierfür ist *cFoxAppl.nFilterBehavior*=0 einzustellen. Die Formulareigenschaft *nFilterBehavior* kann auf 1 eingestellt werden, um das zu VFX 9.0 kompatible Verhalten einzustellen. Mit dem Wert 2 wird der neue Suchdialog aktiviert.

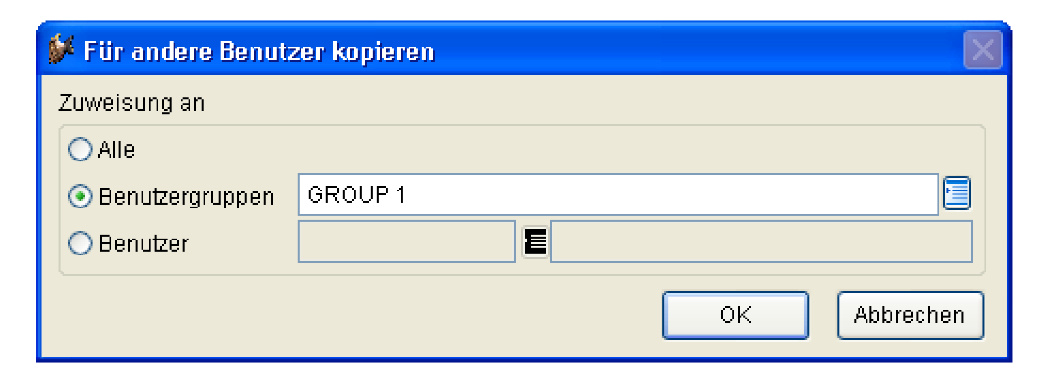
Im neuen Suchdialog stehen die gleichen Filteroptionen wie in VFX 9.0 zur Verfügung. Zusätzlich können Benutzer die Einstellungen speichern und anderen Benutzern oder Benutzergruppen zur Verfügung stellen.



Eine Filtereinstellung kann allen Benutzern, einer Benutzergruppe oder einem Benutzer zugänglich gemacht werden. Jeder Filtereinstellung kann ein Name und eine Beschreibung gegeben werden. Die Filtereinstellungen werden zum aufrufenden Formular gespeichert und können später wieder verwendet werden.

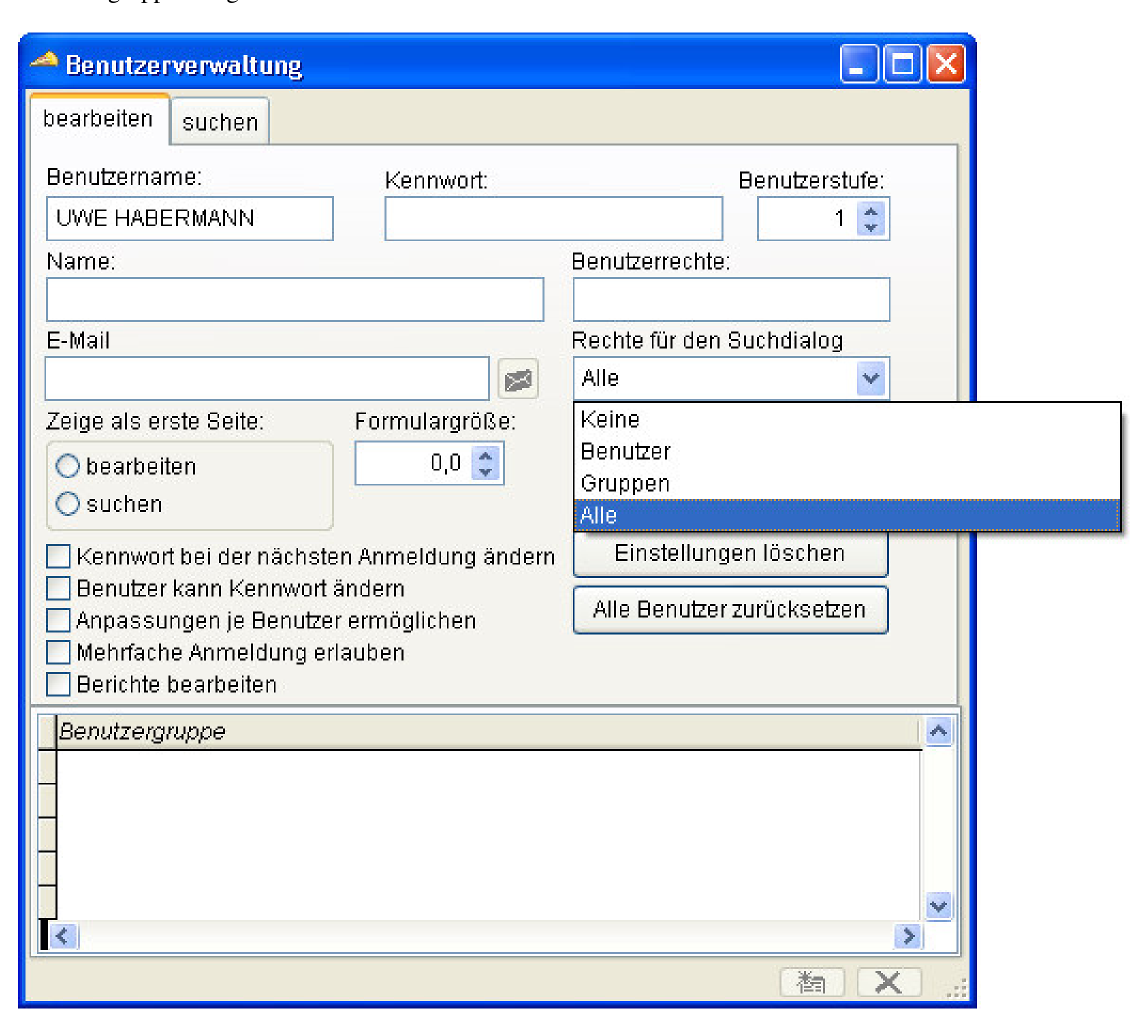
Ein Benutzer kann seine eigenen Filtereinstellungen sehen sowie die Filtereinstellungen, die für alle Benutzer oder für Benutzergruppen freigegeben sind, in denen der Benutzer Mitglied ist. Für jedes Formular kann im Rechtedialog bzw. in der Verwaltung der Benutzergruppen eingestellt werden, welche Benutzerstufe erforderlich ist, um Filtereinstellungen bearbeiten zu können.

Benutzer können Filtereinstellungen neu anlegen, kopieren, für andere Benutzer kopieren, bearbeiten und löschen. Wenn eine Filtereinstellung für andere Benutzer kopiert werden soll, erscheint ein Dialog zur Auswahl des Benutzers bzw. der Benutzergruppe.



Wenn ein eingegebener Filter zu einer leeren Ergebnismenge führt, wird dies dem Benutzer in einer Messagebox angezeigt und der Suchdialog bleibt geöffnet. Wenn alle Filtereinstellungen gelöscht werden, bleibt der Suchdialog ebenfalls geöffnet. Einzelne Zeilen im Suchdialog können jetzt über eine Schaltfläche gelöscht werden.

Die Rechte für den erweiterten Suchdialog können in der Benutzerverwaltung und in der Verwaltung der Benutzergruppen eingestellt werden.



Für jeden Benutzer bzw. für jede Benutzergruppe kann eingestellt werden:

- Keine – Der Benutzer kann Filterbedingungen nicht speichern. Er kann aber Filterbedingungen verwenden, die von anderen Benutzern gespeichert wurden. Dies ist die Standardeinstellung.

- Benutzer – Der Benutzer kann Filterbedingungen zu seiner eigenen Verwendung speichern. Er kann seine Filterbedingungen nicht anderen Benutzern zur Verfügung stellen.

- Gruppen – Der Benutzer kann Filterbedingungen für sich und für Benutzergruppen speichern.

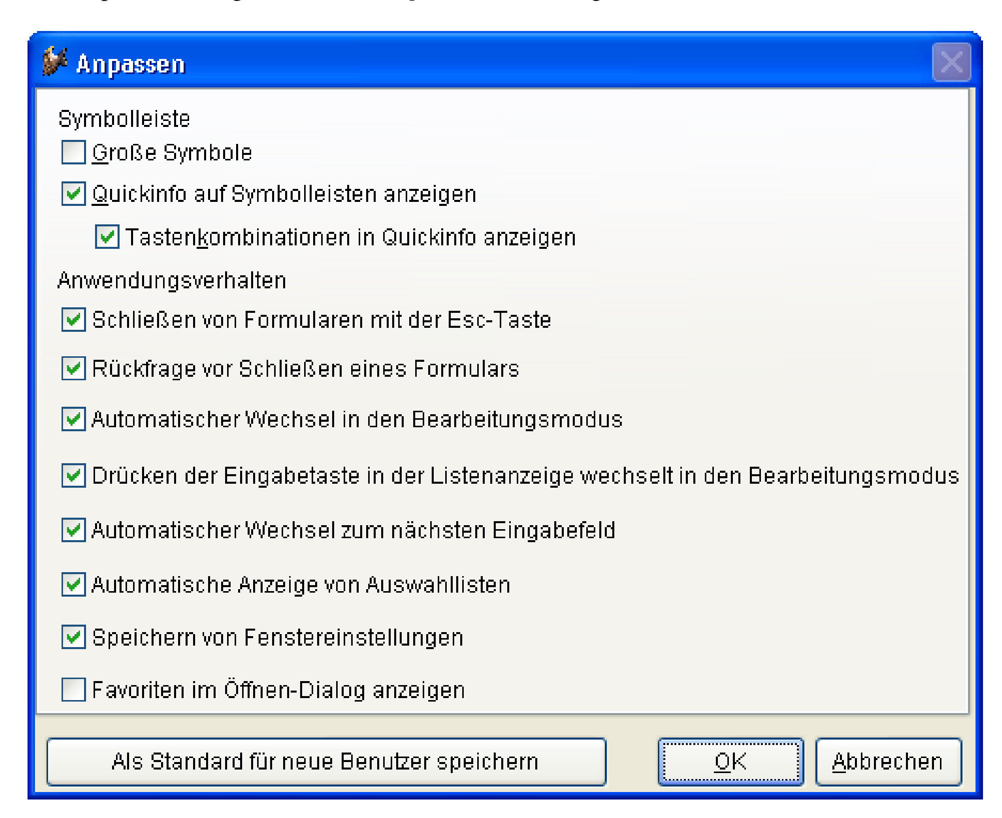
- Alle – Der Benutzer kann Filtereintellungen für jeden speichern.

Wenn ein Benutzer Mitglied in mehreren Gruppen ist, gilt das höchste Recht.

20.10. Anpassen-Dialog

Viele Eigenschaften der Anwendung kann sich jeder Benutzer individuell selbst anpassen. Die Anpassbarkeit dieser Einstellungen kann über die Eigenschaft *lAllowUserCustomization* des Anwendungsobjekts für die Anwendung gesteuert werden. Wenn der Wert dieser Eigenschaft .F. ist, ist der Anpassen-Dialog in der Anwendung nicht sichtbar. Wenn der Wert dieser Eigenschaft .T. ist, kann der Administrator für jeden Benutzer individuell erlauben den Anpassen-Dialog zu verwenden.

Der Anpassen-Dialog kann aus dem *Optionen*-Menü aufgerufen werden.



Der Anpassen-Dialog kann nur von Benutzern angezeigt werden, die das Recht *Anpassungen je Benutzer ermöglichen* haben. Dieses Recht kann nur von Administratoren vergeben werden, wenn die Eigenschaft *goProgram.lAllowUserCustomization* auf .T. eingestellt ist.

Über den Anpassen-Dialog kann jeder Benutzer seine individuellen Einstellungen zu Symbolleisten und zum Anwendungsverhalten machen.

Die Schaltfläche *Als Standard für neue Benutzer speichern* ist nur für Administratoren sichtbar. Wenn ein Administrator auf diese Schaltfläche klickt, werden die aktuell sichtbaren Einstellungen als Standardwerte für neu anzulegende Benutzer gespeichert. Wenn sich ein neuer Benutzer erstmalig anmeldet, gelten diese Standardwerte.

Mit der Schaltfläche *OK* werden die aktuell sichtbaren Einstellungen für den angemeldeten Benutzer übernommen. Mit der Schaltfläche *Abbrechen* werden die Einstellungen verworfen.

20.11. Die Klasse CArchive

Dieser Klasse wurde die neue Eigenschaft *lQuietMode* hinzugefügt. Über diese Eigenschaft kann gesteuert werden, ob alle Anzeigen während eines Archivierungsvorgangs unterdrückt werden sollen.

lQuietMode – Gibt an, ob die Archivierung ohne Meldungen ausgeführt werden soll. Diese Eigenschaft muss eingestellt werden, bevor eine der Methoden *CreateArchive()* oder *ExtractFromArchive()* ausgeführt wird.
.T. – Meldungen unterdrücken (unterdrückt auch die Fortschrittsanzeige)
.F. – Meldungen anzeigen

20.12. Die neue Klasse CTextSkype

Die neue Klasse *CTextSkype* befindet sich in der Klassenbibliothek *Vfxctrl.vcx*. Diese Klasse besteht aus einem Container mit einer Textbox und einer Schaltfläche.



Wenn der Benutzer zur Laufzeit auf die Schaltfläche klickt, wird der in der Textbox befindliche Wert als Skype-Name an das Programm Skype übergeben. Mit Skype ist es möglich Telefongespräche über das Internet zu führen und Sofortnachrichten zu senden. Mehr Informationen zu Skype finden Sie im Internet auf http://www.skype.com.

20.13. Behandlung von Laufzeitfehlern

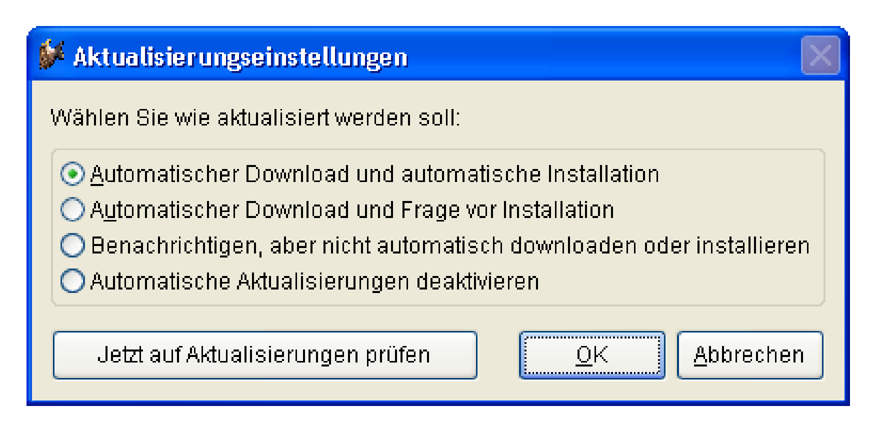
Wenn ein Laufzeitfehler auftritt, prüft die Funktion *OnError* ob eine Verbindung zur Remote Datenbank besteht. Wenn möglich, werden die Fehlerinformationen in der Tabelle *Vfxlog* in der Remote Datenbank gespeichert. Wenn eine Verbindung zur Remote Datenbank nicht möglich ist, werden die Fehlerinformationen in der lokalen Tabelle *Vfxlog.dbf* gespeichert.

Wenn eine VFP-Datenbank verwendet wird, werden die Fehlerinformationen immer in der lokalen Tabelle *Vfxlog.dbf* gespeichert.

Wenn ein Fehler beim Speichern eines Datensatzes auftritt, wird jetzt zusätzlich der Name des Arbeitsbereichs, in dem der Fehler aufgetreten ist, im Fehlerprotokoll gespeichert. Dies hilft bei der Lokalisierung des Problems.

20.14. Aktualisierung der Anwendung

Mit VFX erstellte Anwendungen können über das Internet aktualisiert werden. Wenn die Eigenschaft *lAllowUpdates* des Anwendungsobjekts auf *.T.* eingestellt ist, ist die Aktualisierungsfunktion in der Anwendung aktiviert und die Menüpunkte *Aktualisierung der Anwendung* und *Aktualisierungseinstellungen* stehen zur Verfügung.



Im Dialog *Aktualisierungseinstellungen* können Benutzer zwischen vier Optionen wählen. Wenn automatische Aktualisierungen nicht deaktiviert sind, prüft die Anwendung täglich bei jedem ersten Start ob Aktualisierungen vorliegen.

Die Aktualisierungsfunktionen sind in den Klassen *cUpdate* und *cUpdateEngine* implementiert.

20.14.1. Aktualisierung der Datenbank beim Kunden

Die Aktualisierung der Datenbank beim Kunden wurde erweitert. Das neue Verfahren ist kompatibel zu bisherigen Versionen von VFX. Die Aktualisierung kann also weiterhin dadurch erfolgen, dass eine neue Datenbankstruktur im *Update*-Ordner an den Kunden ausgeliefert wird.

Um das neue Verfahren zu nutzen, muss die Exe-Datei mir einer Versionsnummer versehen werden. Es empfiehlt sich die Versionsnummer automatisch bei jedem Build von VFP erhöhen zu lassen. Bei jedem Erstellen einer Exe-Datei werden über einen Project Hook für alle Datenbanktypen aus der Datei *Config.vfx* im Projektordner Metadaten erstellt. Die Metadaten werden in die Exe-Datei eingeschlossen und stehen so beim Kunden zur Verfügung.

Wenn eine Exe-Datei gestartet wird, wird die Versionsnummer mit der Nummer verglichen, die im Feld *AppVersion* in der Tabelle *Vfxsys* gespeichert ist. Wenn die Version der aktuellen Exe-Datei größer als die gespeicherte Versionsnummer ist, wird die Aktualisierung der Datenbankstruktur gestartet. Die Aktualisierung wird für jede Datenbank durchgeführt, die in *Config.vfx* oder in *Vfxpath.dbf* eingetragen ist. Es werden dabei VFP-Datenbanken und Remote Datenbanken aktualisiert. Anschließend wird die neue Versionsnummer im Feld *AppVersion* in der Tabelle *Vfxsys* gespeichert.

20.15. Datenbankreparatur

Die Datenbankreparaturfunktion von VFX 9.0 wurde erheblich verbessert. Um eine Datenbank reparieren zu können muss die Struktur der korrekten Datenbank bekannt sein. Zu diesem Zweck führt der Project Hook, der in allen VFX-Projekten aktiv ist, bei jedem Erstellen einer Anwendung das Programm *Gendbc.prg* aus. *Gendbc.prg* wird mit VFP geliefert. Dieses Programm liest einen Datenbank-Container aus und generiert dabei eine Programmdatei. Die spätere Ausführung dieser Programmdatei führt dazu, dass eine neue, leere Datenbank mit der gleichen Struktur hergestellt wird. Diese Datenbankstruktur kann für die Reparatur einer beschädigten Datenbank verwendet werden.

Die von *Gendbc.prg* erzeugte Programmdatei wird automatisch in das Projekt eingeschlossen und steht somit zur Ausführung in einer Exe-Datei zur Verfügung. Wenn eine Datenbank gespeicherte Prozeduren enthält, werden diese von *Gendbc.prg* in ein Memofeld einer Tabelle kopiert. Diese generierte Tabelle wird ebenfalls automatisch in das Projekt eingeschlossen.

Ein defekter DBC kann in den meisten Fällen sofort beim Start der Exe-Datei erkannt werden und wird dann ohne Fragen an den Benutzer sicher repariert.

Zur Reparatur aller weiteren Beschädigungen muss manuell aus dem Menü Extras, Datenbankwartung aufgerufen werden.

Diese Probleme lassen sich über den Dialog Datenbankwartung beheben:

- Beschädigte Tabellenköpfe, insbesondere auch defekte Satzzähler, werden in jedem Fall ohne Datenverlust wiederhergestellt.
- Defekte Datensätze, zum Beispiel mehrfach vorkommende Primärschlüssel, können sicher erkannt werden und die doppelten Datensätze werden automatisch gelöscht. Anschließend kann die Tabelle wieder geöffnet und verwendet werden. Es ist sicher, dass nur zerstörte Datensätze verloren gehen.
- Fehlende DBF-Dateien können wiederhergestellt werden. Die Daten der betreffenden Tabelle sind dann natürlich verloren.
- Fehlende CDX-Dateien werden in jedem Fall ohne Datenverlust wiederhergestellt.
- Beschädigte CDX-Dateien können in vielen Fällen erkannt und ohne Datenverlust wiederhergestellt werden.
- Defekte Memofelder können in fast allen Fällen repariert werden. Die Memos des betreffenden Datensatzes gehen dabei verloren.
- Fehlende Memo-Dateien können wiederhergestellt werden. Dabei gehen natürlich alle Memos der betreffenden Tabelle verloren.

Tabellen können repariert werden, während der DBC von anderen Benutzern geöffnet ist. Zur Reparatur ist in jedem Fall nur der exklusive Zugriff auf die zu reparierenden Dateien erforderlich. Ausgenommen, wenn im Dialog „gesamte Datenbank“ gewählt wird. Hierzu ist der exklusive Zugriff auf alle Dateien inklusiv DBC erforderlich.

20.16. Bessere Unterstützung von geringen Farbtiefen

Bei einer Farbtiefe von maximal 256 Farben werden im XP-Öffnen-Dialog automatisch Bitmap-Dateien verwendet, die bei geringer Farbtiefe optisch ansprechend dargestellt werden.

20.17. Terminalserver-Unterstützung

Wenn eine Anwendung in einer Terminalserver-Sitzung läuft, wird automatisch die Bitmap-Anzeige mit dem VFP-Befehl *SYS(602)* optimiert.

Zur weiteren Optimierung der Darstellung von Symbolleisten in Anwendungen, die in Terminalserver-Sitzungen laufen sollen, empfehlen wir der *Visible* Eigenschaft von Steuerelementen nur dann Werte zuzuweisen, wenn dies unbedingt erforderlich ist. Dadurch werden unnötige ausgeführte *REFRESH()*Ereignisse vermieden und die Symbolleiste wird flackerfrei angezeigt.

Beispiel-Code für das *REFRESH()* Ereignis in *cAppToolBar* oder *cAppNavBar*:

```
* Nicht empfehlenswert:
DODEFAULT()
This.cmdNew.Visible = .F.

* Empfehlenswert:
DODEFAULT()
IF This.cmdNew.Visible
   This.cmdNew.Visible = .F.
ENDIF
```

20.18. Weitere Verbesserungen für Endbenutzer

Alle benutzerspezifischen Einstellungen, wie Formulargröße, Position auf dem Bildschirm und Grid-Einstellungen können jetzt wahlweise entsprechend der verwendeten Bildschirmauslösung gespeichert und geladen werden. Dafür ist die Eigenschaft *lSaveFormLayoutResolutionDependent* in der Klasse *cFoxAppl* in der Klassenbibliothek *Appl.vcx* auf *.T.* zu stellen. Der Standardwert ist *.F.* Diese Einstellung kann auch mit dem VFX – Application Builder gemacht werden.

Die Archivierungsfunktion aus dem Menü erstellt jetzt Dateinamen, die aus dem Ordnernamen, dem Datenbanknamen sowie dem aktuellen Datum im ANSI-Format bestehen.

In Formularen mit einem Treeview-Steuerelement erfolgt die Navigation mit den Schaltflächen vor, zurück, Anfang und Ende in der Symbolleiste, jetzt entsprechend der logischen Anzeigefolge im Treeview-Steuerelement.

Die Daten aus allen Grids können per Drag & Drop in andere Anwendungen gezogen werden. Dieses Verhalten ist global und je Grid einstellbar. Wenn die Eigenschaft *nOLEDragGrid* der Klasse *cFoxAppl* in der Klassenbibliothek *Appl.vcx* auf *1* eingestellt wird, können die Daten aus allen Grids der Anwendung per Drag & Drop in andere Anwendungen gezogen werden. Dies war das Standardverhalten. Wenn der Wert dieser Eigenschaft auf *0* eingestellt wird, kann dies für jedes Grid individuell mit der Eigenschaft *lOLEDragGrid* eingestellt werden. *1* ist der Standardwert. Wenn der Wert der Eigenschaft *nOLEDragGrid* auf *2* eingestellt wird, ist OLE Drag & Drop in allen Grids der Anwendung ausgeschaltet.

Bereits in VFX 9.0 konnte mit der Eigenschaft *lKeepLoggedUsers* des Anwendungsobjekts eingestellt werden, ob Benutzeranmeldungen protokolliert werden sollen. Für eine Anwendung konnte mit dem VFX – Application Builder eingestellt werden, ob sich Benutzer mehrmals gleichzeitig anmelden dürfen. Der Administrator kann für jeden Benutzer individuell einstellen, ob eine mehrfache Anmeldung erlaubt ist. Zu diesem Zweck gibt es in der Benutzerverwaltung das Kontrollkästchen *Mehrfache Anmeldung erlauben*.

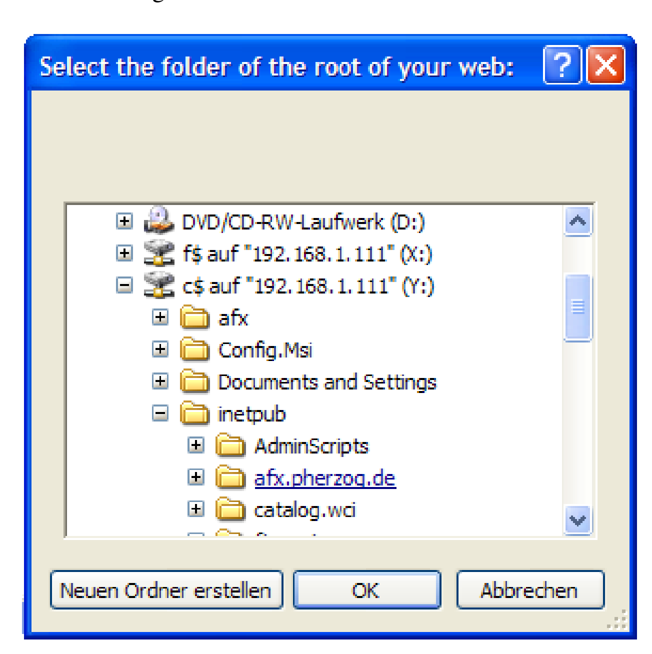
Das Menü *Hilfe* in Anwendungen wurde erweitert. Über den Eintrag *Besuchen Sie unsere Website* kann der Benutzer die Website besuchen, deren URL in der Eigenschaft *cCompanyWebSiteURL* des Anwendungsobjekts hinterlegt ist. Über den Menüpunkt *So erreichen Sie uns* können dem Benutzer Kontaktinformationen angezeigt werden. Als Kontaktinformation dient eine HTML Datei, die in der Tabelle *Vfxinternfiles.dbf* im Datensatz mit *type="contactus"* gespeichert ist.

21. Anhang I - VFX AFX Wizard

von Peter Herzog

Mit dem VFXAFXWizard können sie VFXMasken, welche mit dem Form Wizard erzeugt wurden, in Internetfähige DHMTL Masken umwandeln.

Sobald Sie den Wizard das erste Mal starten, werden Sie aufgefordert das Ausgabeverzeichnis für die erzeugten Dateien anzugeben.

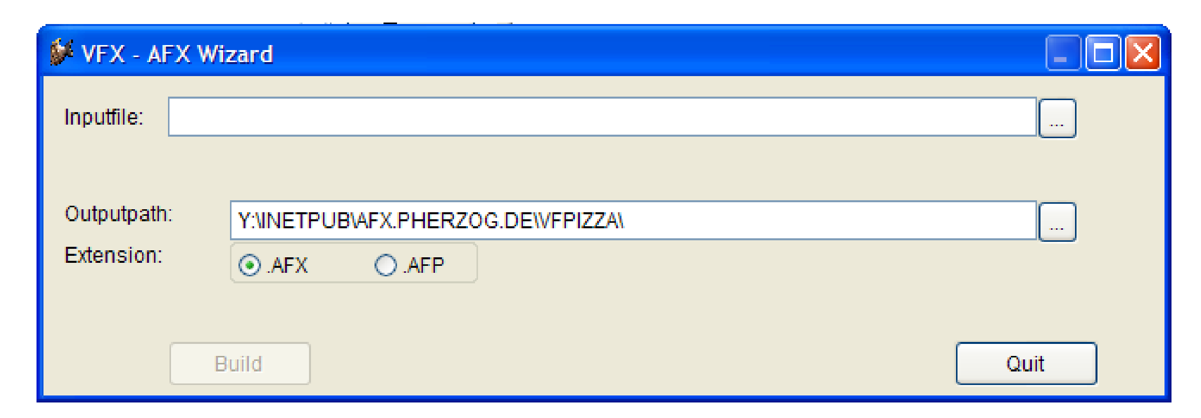


Sollten Sie bereits im internen Netzwerk einen Internetserver besitzen, so können Sie bereits hier den Pfad angeben, in dem Ihre Daten abgespeichert werden sollen. (HKLM\SOFTWARE\Microsoft\InetStp)
Der Wizard sucht sich in der Registry den Pfad eines eventuell lokal installierten IIS und schlägt diesen Pfad bereits vor.

Nun werden alle notwendigen Dateien, wie Bilder, Stylesheets, vorgefertigte HTML-Seiten unter
`C:\Dokumente und Einstellungen\All Users\Anwendungsdaten\dFPUG\Visual Extend\11.0\Wizfiles`
angelegt.

Und es wird die Metadatentabelle VFXAFXMETA.DBF unter
`C:\Dokumente und Einstellungen\All Users\Anwendungsdaten\dFPUG\Visual Extend\11.0`
erzeugt.

Nun erscheint die Maske des Wizards.



Der vorher ausgewählte Pfad ist als Outputpath: voreingestellt. Jede Änderung wird in der VFXAFXMETA.DBF gespeichert.

Es kann hier gewählt werden, ob .AFX als Extension verwendet werden soll, oder .AFP
Der erzeugte Code ist identisch, da beide Script Engines gleichermaßen den Code abarbeiten können.

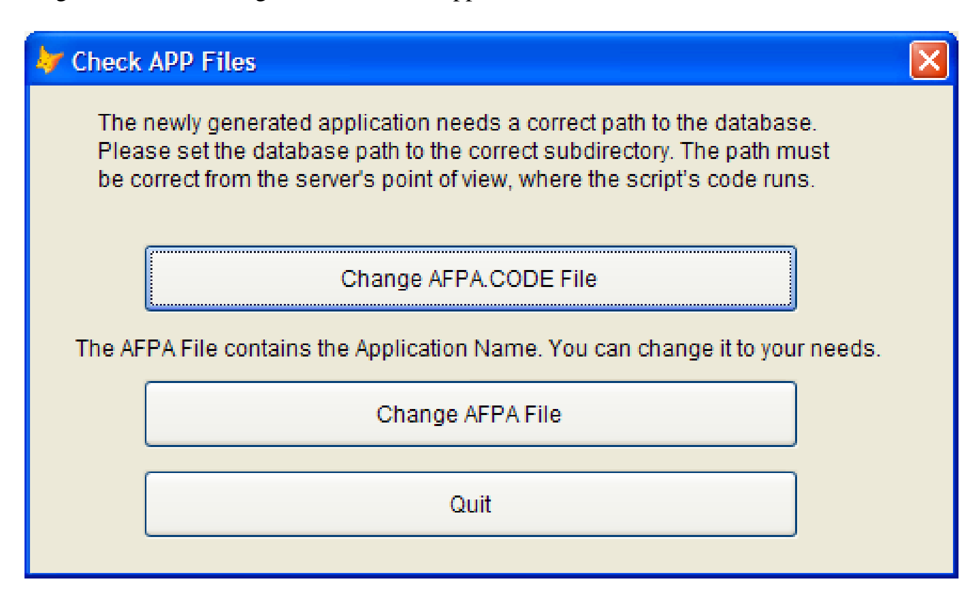
Sobald Sie eine Maske aus Ihrem Projekt ausgewählt haben und durch einen Klick auf „Build" den Wizard starten, wird im Hintergrund die komplette Umgebung geladen, Dateien geöffnet und eventuelle SQL-Server anfragen ausgeführt.

Sie müssen sich an der laufenden Applikation auch anmelden, damit die Form mit den notwendigen Rechten geöffnet werden kann.

Nun wird die laufende Form Objekt für Objekt analysiert und mit Hilfe der Meta-Tabelle eine lauffähige HTML-Seite erzeugt.

Bei jedem Aufruf wird überprüft ob unterhalb des Ausgabepfades die notwendigen Zusatzdateien vorhanden sind. Sollte eine Datei fehlen, so wird sie aus
`C:\Dokumente und Einstellungen\All Users\Anwendungsdaten\dFPUG\Visual Extend\11.0\Wizfiles`
geholt.

Nachdem alle Dateien erzeugt wurden und auch die .AFPA bzw. .AFPA.CODE erzeugt wurden, werden Sie aufgefordert, den richtigen Pfad und den Applikationsnamen aus Sicht des Servers einzustellen.



Achten Sie auf folgenden Codeteil:

```
with goprogram
   .oConnMgr = Createobject("cConnectionMgr")
   .SetupDataAccessProps(.f.,1)
   .cdatadir="c:\vfx110traders\data\"
   .cmaindatabase="tastrade.DBC "
   .cvfxdir=justpath(File.cLocation)
   .lautologin = .f.
   .ldebugmode = .f.
   .cdatasourcetype = "Native"
Endwith
```

Hier muss der Pfad unter .cdatadir und die Datenbank unter .cmaindatabase eingestellt werden. Unter Umständen müssen Sie die notwendigen Dateien auch noch auf den Server kopieren.

Abschließend wird in der vfxfopen.dbf bei der generierten Maske das Feld inetlevel auf 1 gesetzt. Erst dadurch ist es möglich in xpopendir.af(x/p) die Maske als Link aufzurufen.

21.1. Wichtiger Hinweis

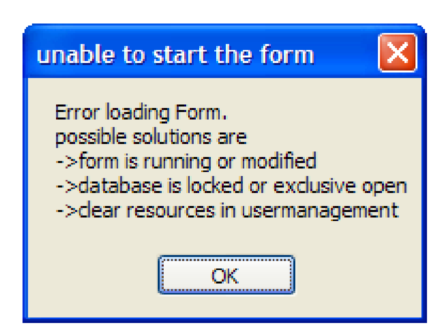
Achten Sie immer darauf, dass der Pfad **aus Sicht des Servers** einzustellen ist. Dies bedeutet, das z.B.: das Rootverzeichnis des Webservers meistens auf dem Laufwerk C liegt. Sollten Sie ein Laufwerk auf dem Server gemapped haben, so ist dies ein anderer Laufwerksbuchstabe. Da die AFX oder AFP aber den Server als „Arbeitsplatz“ sehen, ist das Laufwerk C für sie die Hauptpartition. Achten Sie außerdem darauf, dass die Daten möglichst nicht unterhalb der HTML Seiten liegen, da sie ansonsten unter Umständen aus dem Internet heraus lesbar sind.

Wird ein Server neu installiert, so liegt das Rootverzeichnis unter c:\inetpub\wwwroot. Und wird dem Server nun die Domaine www.meinedomain.de zugewiesen, so wird per default die Datei c:\inetpub\wwwroot\default.htm gelesen. Wenn nun die Daten unterhalb von c:\inetpub\wwwroot gespeichert werden, sind diese unter Umständen direkt aus dem Internet heraus ansprechbar und werden, wenn nicht extra gesichert, sogar über das Internet ladbar.

Sie sollten daher die Daten immer außerhalb dieses Verzeichnisses speichern. Nun aber müssen sie aber auch darauf achten, dass die AFX oder AFP auf die Daten zugreifen können.
Beachten Sie dabei die Zugriffsrechte.

21.2. Mögliche Probleme beim Erzeugen einer Internetform:

Unter Umständen erhalten Sie folgende Fehlermeldung:



Der häufigste Fall ist, dass die Form gerade zum Bearbeiten geöffnet ist.

Eine weitere Fehlerquelle, dass man die Daten unter Umständen in einer 2ten VFP9 Umgebung exklusiv geöffnet hat.

Und unter Umständen kann es vorkommen, dass sie zuerst die Applikation starten müssen und unter Menüpunkt Extras/Benutzerverwaltung „Einstellungen löschen“ klicken müssen.

21.3. Wie arbeitet der VFX AFX Wizard

Sobald im Wizard auf [Build] geklickt wird, wird die vorher ausgewählte Maske mit der kompletten Applikation und Ihren Daten geöffnet.

Diese „lebende" Form wird nun analysiert und die Daten werden in einem internen Cursor gespeichert.

Auch für jedes Grid wird ein extra Cursor intern angelegt.

Sobald die Daten gesammelt wurden, wird die Applikation bzw. die Form wieder geschlossen.

Nun greift der Wizard auf die VFXAFXMETA.DBF zurück und verarbeitet jeden Datensatz des zuvor angelegten Cursors.

Ausschlaggebend ist der Klassenname, welcher herangezogen wird, um in der vfxafxmeta nach dem richtigen Datensatz zu suchen.

Wird der Klassenname gefunden, und ist der hinterlegte Code ***nicht*** leer, wird der Inhalt von cmemo mittels Textmerge Befehl verarbeitet und im HTML-Code eingefügt.

Wenn der Klassenname nicht gefunden wird, wird er automatisch in der vfxafxmeta.dbf angelegt und der Datensatz der Basisklasse wird gesucht. Nun wird dieser anstelle des Klassennamens verwendet.

Sie können diese Art der „Vererbung" aber auch absichtlich unterbrechen, was z.B. im *ctoolbarbutton* gemacht wird. Für die Speedbar, bzw. die normale Toolbar in den VFX-Formularen wird eine extra Klasse verwendet und somit auch ein extra Code, welcher im HTML eingefügt wird.
Jeder Toolbarbutton hat den Klassennamen „*ctoolbarbutton*". Da in einer Form, welche mit der Speedbar ausgestattet ist, auch der Toolbarbutton vorkommt und dieser auf der Basisklasse „textbox" basiert, würde im erzeugten HTML eine Reihe von Buttons erscheinen, weil ctoolbarbutton nicht verwendet wird und die Basisklasse „textbox" anstelle dessen im HTML eingefügt wird.

Dies wird aber explizit verhindert, indem in cmemo von „ctoolbarbutton" der Text
<!-- disabled-->
eingefügt ist. Sie sehen, dass es sich um einen HTML Kommentar handelt. Dieser wird nun anstelle des Codes für die Basisklasse „textbox" eingefügt und es sind keine Buttons mehr im HTML zu sehen.

HTML funktioniert immer mit einem Öffnendem und einem Schließendem „TAG". Also jedes <div> muss mit einem </div> wieder geschlossen werden.
Deshalb gibt es für jeden Klassen-Datensatz und Basisklassen-Datensatz ein Pendant zum schließen. Beispiel textbox -> textbox_end
Sinn macht es bei Pageframe, Page, cdataformpage also allen Containerobjekten. Jeder Container beinhaltet mehrere andere Objekte und muss deshalb am Ende wieder geschlossen werden. Beispiel Pageframe_end, welches nur ein </div> beinhaltet.

21.4. Die Variablen mit den Daten der Form

Wie bereits beschrieben, wird intern ein Cursor gehalten, welcher alle notwendigen Daten des originalen Formulars beinhaltet.

Diese Daten sind direkt als Variablen ansprechbar und müssen in doppelten Spitzen Klammern eingefügt werden.

Beispiel Optiongroup:

```
<div id="div_<<cname>>_<<nlfd>>" style="position: absolute; border-
style:solid ;border-width:0px ;
left:<<nleft>>; top:<<ntop>> ; height: <<nheight>> ; width: <<nwidth>>
;z-index:<<nlevel>>">
```

cname, nlfd, nleft, ntop, nheight, nwidth und nlevel sind Variablen, aus dem erwähnten Cursor und werden direkt als Werte im erzeugten HTML eingefügt.

Am Beispiel der textbox sehen Sie das auch komplexe Ausdrücke eingefügt werden können:

```
size="<<int(IIF(LEN(cinputmask)>0,LEN(cinputmask),nwidth/FONTMETRIC(6,cf
ont,nfontsize)))>>"
```

Unter Umständen ist es jedoch sinnvoller kompletten VFP Code auszuführen. Dafür gibt es in der vfxafxmeta.dbf ein Flag mit dem Namen lcode. Ist dies auf .T. wird der Inhalt von cmemo temporär compiliert und ausgeführt.

Ein Beispiel dafür finden Sie unter Pagescript indem der Javascriptcode erzeugt wird.

```
local lcs
lcs=[<script>]+chr(13)+chr(10)
lcs=lcs+[function activate]+alltrim(conload)+[()] +chr(13)+chr(10)
lcs=lcs+[{]+chr(13)+chr(10)
for i=1 to nmaxcount
   if i=nlfd
lcs=lcs+[Page]+trans(nlevel)+[_]+trans(i)+[.style.visibility="visible";]
+chr(13)+chr(10)
lcs=lcs+[changetab(Tabspan]+trans(nlevel)+[_]+trans(i)+[,"vfxafximage/ta
b-active.png");]+chr(13)+chr(10)
lcs=lcs+[Tabspan]+trans(nlevel)+[_]+trans(i)+[.onmouseout='changetab(Tab
span]+trans(nlevel)+[_]+trans(i)+[,"/vfxafximage/tab-
active.png");']+chr(13)+chr(10)
   else
lcs=lcs+[Page]+trans(nlevel)+[_]+trans(i)+[.style.visibility="hidden";]+
chr(13)+chr(10)
lcs=lcs+[changetab(Tabspan]+trans(nlevel)+[_]+trans(i)+[,"vfxafximage/ta
b.png");]+chr(13)+chr(10)
lcs=lcs+[Tabspan]+trans(nlevel)+[_]+trans(i)+[.onmouseout='changetab(Tab
span]+trans(nlevel)+[_]+trans(i)+[,"/vfxafximage/tab.png");']+chr(13)+ch
r(10)
   endif
endfor
lcs=lcs+[};]+chr(13)+chr(10)
lcs=lcs+[</script>]+chr(13)+chr(10)
return lcs
```

Sollte ein Fehler auftreten, wenn solch ein Code ausgeführt wird, wird der Fehler als HTML-Kommentar im HTML gespeichert.

Wenn Sie also Quellcode selbst verändern oder erstellen, so überprüfen Sie immer das erzeugte Ergebnis.

21.5. Die Laufzeittabellen

Der Cursorname der VFXAFXMETA.DBF zur Laufzeit lautet htmlbuildx

Ckey c(40)	Der Klassenname
Cdesc c(80)	Beschreibung
Cmemo M	Inhalt als HTML oder als Code
Lparam l	.T. wenn Steuerdaten für den Wizard
Lcode l	Bei .T. wird es als VFP-Code compiliert
Nvers n(5,2)	Versionsnummer bei 99.99 schreibgeschützt

In der Tabelle sind auch noch Steuerdaten für den Wizard selbst abgelegt. Diese Steuerdaten sind mit lparam = .T. gekennzeichnet und lauten:

Extension	.AFX oder .AFP wird direkt in der Wizardmaske gesetzt
Outputpath	Ausgabepfad der erzeugten Dateien
Postfix	Namenserweiterung nach dem Dateinamen
Postfixexec	Namenserweiterung des Ausführenden Codes "_EXEC"
Postfixproc	Namenserweiterung der Proceduredatei "_PROC"
Prefix	Vorangestellte Zeichenkette, vor jeder Datei "VFX_"

Der Cursorname der Form zur Laufzeit lautet htmltemp
Diese Tabelle sollte nur innerhalb eines lcode=.T. verwendet werden. Alle notwendigen Felder werden zusätzlich in Public Variablen hinterlegt. Siehe weiter unten.

level i	Der Level für DHTML Ebenen
name c(220)	Der Name der Klasse, wie er vom Wizard erzeugt wird. Dieser Name beinhaltet immer auch alle Parentnamen.
namesort c(220)	Ein Sortierfeld, nach dem dann abgearbeitet wird.
baseclass c(20)	Die Basisklasse jeden Objektes
class c(20)	Der Klassenname jeden Objektes
parent m	Der Parent jedes Objektes
caption c(100)	
left i	
top i	
width i	
height i	
font c(30)	
fontsize i	
forecolor i	
backcolor i	
alignment i	
value m	Wird unter Umständen anders verwendet. Cursor: value beinhaltet den Alias Cursoradapter: value beinhaltet den Alias Grid: value beinhaltet controlsource der Column[x]
csource m	Controlsource Grid: csource beinhaltet recordsource Textbox: ist thisform als text vorhanden, wird es in g_thisform_ gewandelt. Notwendig für Viewparameter
backstyle i	
lfd i	Page: lfd enthält Pageorder Onload wird ebenfalls damit gefüllt
maxcount i	Page: beinhaltet Pagecount
inputmask m	
tabs l	Pageframe.tabs
visible l	
onload c(40)	Scriptcode: "Page_"+TRANSFORM(nlevel)+"_"+TRANSFORM(lfd)
tablen i	20+(5*LEN(ALLTRIM(caption)))
tableft i	Addierte Tablen
speedbar l	.T. wenn speedbar in der Form verwendet
RowSource m	
RowSourceType i	
ColumnCount i	
BoundColumn i	
tabindex i	
pageframeindex i	Fortlaufende Nummer der vorhandenen Pageframes
valid M	Inhalt aus afx_valid
gotfocus M	Inhalt aus afx_gotfocus
lostfocus M	Inhalt aus afx_lostfocus
keypress M	Inhalt aus afx_keypress
click M	Inhalt aus afx_click
dblclick M	Inhalt aus afx_dblclick
tooltiptext M	
statusbartext M	
user M	Noch nicht verwendet
cviewparameter c(100)	

Es werden Public Variablen zur Verfügung gestellt, welche die Daten des jeweiligen Datensatzes beinhalten. Diese sollten verwendet werden, anstelle des Feldes aus dem Cursor.

Es wird als Beispiel bei cforecolor der originalwert automatisch in das HTML-pendant umgewandelt.

ckey	ALLTRIM(htmlbuildx.ckey)
cdesc	ALLTRIM(htmlbuildx.cdesc)
nlevel	htmltemp.level
cname	ALLTRIM(htmltemp.name)
cbaseclass	ALLTRIM(htmltemp.baseclass)
ccaption	ALLTRIM(htmltemp.caption)
nleft	htmltemp.left
ntop	htmltemp.top
nwidth	htmltemp.width
nheight	htmltemp.height
cfont	ALLTRIM(htmltemp.font)
nfontsize	htmltemp.fontsize
nforecolor	htmltemp.forecolor
nbackcolor	htmltemp.backcolor
cforecolor	ALLTRIM(ohtmlbuilder.htmlcolor(htmltemp.forecolor))
cbackcolor	ALLTRIM(ohtmlbuilder.htmlcolor(htmltemp.backcolor))
nalignment	htmltemp.alignment
cvalue	ALLTRIM(htmltemp.value)
ccontrolsource	ALLTRIM(htmltemp.csource)
Cvalid	htmltemp.valid
Cgotfocus	htmltemp.gotfocus
Clostfocus	htmltemp.lostfocus
Cclick	htmltemp.click
Cdblclick	htmltemp.dblclick
Ckeypress	htmltemp.keypress
cRowSource	htmltemp.rowsource
nRowSourceType	htmltemp.rowsourcetype
nColumnCount	htmltemp.columncount
nBoundColumn	htmltemp.boundcolumn
cstatusbartext	htmltemp.statusbartext
Ctooltiptext	htmltemp.tooltiptext
nbackstyle	htmltemp.backstyle
nlfd	htmltemp.lfd
nmaxcount	htmltemp.maxcount
cinputmask	htmltemp.inputmask
ltabs	htmltemp.tabs
lvisible	htmltemp.visible
conload	htmltemp.onload
ntablen	htmltemp.tablen
ntableft	htmltemp.tableft
lspeedbar	htmltemp.speedbar
nRowSource	htmltemp.rowsource
nRowSourceType	htmltemp.rowsourcetype
ntabindex	htmltemp.tabindex
npageframeindex	htmltemp.pageframeindex

Der Cursorname eines Grids zur Laufzeit lautet htmltempgrid<n> n ist die fortlaufende Nummer.

feldnr i	Fortlaufende Nummer
caption c(30)	Überschrift für die Gridcolumn
width i	Breite
csource c(50)	Controlsource
crecordsource c(50)	Recordsource

Es werden außerdem noch folgende Felder als Variablen angelegt, wodurch der Cursor des Grids nicht unbedingtverwendet werden muss.

ngridfeldnr	feldnr
cgridcaption	caption
ngridwidth	width

cgridcsource	ALLTRIM(csource)
Cgridrecordsource	ALLTRIM(crecordsource)
Cgridshortcsource	STRTRAN(ALLTRIM(lower(csource)),ALLTRIM(LOWER(crecordsource))+".","")

Der Cursorname des goprogram-Objektes der Applikation lautet goprogdata

cmaindatabase c(100)	Name der Hauptdatenbank
cdatasourcetype c(100)	Datenquellentyp (Native, ODBC usw.)
clangid c(100)	Sprachid
cmaintitle c(100)	Haupttitel der Applikation

Es werden vor dem Build-Lauf noch globale Variablen gefüllt, welche ebenfalls verwendet werden können.
Diese Variablen sind teilweise direkt als Properties des Wizards ausgelegt.
Es gibt auch noch Methoden, welche direkt verwendet werden können:

Methoden

ohtmlbuilder.evalthis(tcwhat)	Evaluiert jeden Begriff oder jeden Wert. Handelt es sich um eine Zeichenkette, wird sie als ckey in htmlbuildx (vfxafxmeta.dbf) gesucht und der Inhalt von cmemo wird wiederum evaluiert. Dies wird sogar rekursiv vorgenommen.
ohtmlbuilder.htmlcolor(tncolor)	Umwandlung einer RGB-Zahl nach deren Internetentsprechung

Properties

ohtmlbuilder.cappname	VFX-Projekt Verzeichnis
ohtmlbuilder.cappdir	Pfad zur VFXAFXMeta.dbf
ohtmlbuilder.cappfullname	Pfad zum VFX-Projekt
Outputpath	Ausgabepfad
Extension	Gewählte Erweiterung .AFX oder .AFP
Filename	Dateiname des Formulars

21.6. Der Aufbau der erzeugten Dateien

Sobald eine Form mit dem VFX AFX Wizard umgewandelt wird, werden automatisch die folgenden Verzeichnisse und Dateien angelegt. Achtung. Abhängig der Auswahl ob AFX oder AFP erzeugt werden soll, werden dementsprechend die Dateiendung und die Links angepasst.

Das Verzeichnis LIB beinhaltet alle Libraries für die Internetapplikation

```
Lib
Lib \ afx.dll
Lib \ vfx.fll
Lib \ vfxafx.vcx
Lib \ vfxafx.vct
```

Im Verzeichnis Include sind die Headerdateien untergebracht. Es werden alle VFX Headerdateien mitgeliefert, obwohl nicht alle benutzt werden.

```
Lib \ Include
Lib \ Include \ FOXPRO.H
Lib \ Include \ FOXPRO_REPORTING.H
Lib \ Include \ REPORTLISTENERS.H
Lib \ Include \ REPORTLISTENERS_LOCS.H
Lib \ Include \ USERDEF.H
Lib \ Include \ USERMSG.H
Lib \ Include \ USERTXT.H
Lib \ Include \ VFX.H
Lib \ Include \ VFXDEF.H
```

```
Lib \ Include \ VFXGLOBAL.H
Lib \ Include \ VFXMSG.H
Lib \ Include \ VFXOFFCE.H
Lib \ Include \ VFXTOOLBOX.H
Lib \ Include \ VFXTXT.H
Lib \ Include \ _FRXCURSOR.H
```

Im Verzeichnis Program liegt die vfxfunc.prg aus dem VFX95 Projekt.

```
Program
Program \ vfxfunc.prg
Program \ vfcfunc.fxp
```

Die Images findet man unter

```
Vfxafximage
```

Das Grid ist mit einem Javascript Bestandteil ausgestattet. Dieser ist hier untergebracht

```
Vfxafxjs \ grid.js
```

Die Stylesheets

```
Vfxafxstyle \ basic.css
Vfxafxstyle \ grid.css
```

Wird mit Cursoradapter gearbeitet wird die config.vfx in die Datei config.afx umgewandelt. Zum Bearbeiten dieser config.afx können sie das mitgelieferte Programm afxconfig.exe verwenden. Diese Datei wird angelegt, wenn Sie nicht vorhanden ist. Sie wird nicht bei jeder Maske neu erzeugt.

```
config.afx
```

Die Applikations Datei. Diese Datei wird angelegt, wenn Sie nicht vorhanden ist. Sie wird nicht bei jeder Maske neu erzeugt.

```
<app>.AFPA
```

Die Applikations-Code Datei, welche in allen Formularen hinein kompiliert wird. Wichtig sind der Pfad und der Name. Diese Datei wird angelegt, wenn Sie nicht vorhanden ist. Sie wird nicht bei jeder Maske neu erzeugt.

```
<app>.AFPA.CODE
```

Die Include-Datei der Applikation. Hier ist die Klassendefinition des Cursoradapters enthalten. Diese Datei wird angelegt, wenn Sie nicht vorhanden ist. Sie wird nicht bei jeder Maske neu erzeugt.

```
<app>.AFPI
```

Die Loginmaske und die dazu gehörige Login Validierung. Diese Dateien werden aus dem Wizfiles Verzeichnis kopiert.

```
LOGIN.AFP
LOGINVALID.AFP
LOGINVALID.AFP.CODE
```

Die Oberfläche und das Menü für die Applikation. Es wird direkt eine vfxfopen.dbf ausgelesen. Das Feld Inetlevel wird dabei berücksichtigt.

```
XPOPEN.AFP
XPOPEN.AFP.CODE
XPOPENBOTTOM.HTM
XPOPENDIR.AFP
XPOPENDIR.AFP.CODE
XPOPENMAIN.HTM
XPOPENTOP.HTM
```

Die eigentliche Form besteht aus mehren Dateien:

Das eigentliche Formular. Die einzelnen Seiten der Pageframe werden mittels Javascript umgeschalten. Die Grids werden in einem IFrame dargestellt.

```
vfx_<form>.AFP
```

Der Codeteil des Formulars mit Dateiöffnungsroutinen und Filtersetzungen

`vfx_<form>.AFP.CODE`

Eventuelle Definitionen von Cursoradaptern werden hier abgelegt
`vfx_<form>.AFPI`

Der Execute Teil des Formulares. Sobald ein Button geklickt wird, wird hier auf die einzelnen Aktionen reagiert. Unter Umständen wird auf andere Seiten weiter verzweigt, wie z.B. beim Filter
`vfx_<form>_EXEC.AFP`

Der Codeteil des Execute Teils mit Dateiöffnungsroutinen und Filtersetzungen
`vfx_<form>_EXEC.AFP.CODE`

Der Filterdialog
`vfx_<form>_filter.AFP`

Dateiöffnungsroutinen für den Filterdialog
`vfx_<form>_filter.AFP.CODE`

Der Execute Teil des Filters mit Weiterleitung zur original Maske
`vfx_<form>_filter_exec.AFP`

Die Gridmaske
`vfx_<form>_grid<lfd>.AFP`

Dateiöffnungsroutinen für die Gridmaske
`vfx_<form>_grid<lfd>.AFP.CODE`

Die Procedure Datei, in der die AJAX-Codeteile abgearbeitet werden
`vfx_<form>_PROC.AFP`

Dateiöffnungsroutinen für die Procedure Datei
`vfx_<form>_PROC.AFP.CODE`

21.7. AJAX

Ist die Abkürzung für **Asynchrones Java and XML.**
Es bedeutet nichts anderes, als das Asynchron mit Hilfe von Java und XML Daten übertragen werden.

In AFX wird der Code automatisch erzeugt, sobald eine der folgenden Methoden in einer Klasse gefunden werden:

AFX_GotFocus
AFX_LostFocus
AFX_KeyPress
AFX_Valid

In der HTML-Maske werden daraufhin diese Klassen in Javascript-Code umgewandelt.

"AFX_Valid" wird zu "onChange"
"AFX_Keypress" wird zu "onKeypress"
"AFX_Gotfocus" wird zu "onFocus"
"AFX_Lostfocus" wird zu "onBlur"

Die dazugehörigen Scriptteile sind in der VFXAFXMeta.dbf unter den Namen „KeypressCode", „GotfocusCode", „LostfocusCode" und „ValidCode" zu finden.

Es wird z.B. beim ValidCode folgendes Script eingefügt:

```
<script type="text/javascript">
function id_<<cname>>_Valid(Feld) {
   var DataToSend = "controlfield=id_<<cname>>_Valid&"
   DataToSend = DataToSend + "value="+Feld+"&"
   DataToSend = DataToSend + "recno=<%?recno()%>&"
   DataToSend = DataToSend + "alias=<%?alias()%>&"
   DataToSend = DataToSend + "controlsource=<<ccontrolsource>>"
   var xmlhttp = new ActiveXObject("Microsoft.XMLHTTP")
   xmlhttp.Open("POST","<<filename>>_PROC<<extension>>",false)
   xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-
urlencoded")
   xmlhttp.send(DataToSend)
   var xmldoc = new ActiveXObject("Microsoft.XMLDOM")
   xmldoc.async="false"
   xmldoc.loadXML(xmlhttp.responsetext)
   cfooter.innerText
=xmldoc.getElementsByTagName("statustext").item(0).text
   if(xmldoc.getElementsByTagName("error").item(0).text== "1")
   {
      if (xmldoc.getElementsByTagName("message").item(0).text != "")
      {
         alert(xmldoc.getElementsByTagName("message").item(0).text)
      }
      document.<<filename>>.id_<<cname>>.focus()
   }else
   {
      if (xmldoc.getElementsByTagName("message").item(0).text != "")
      {
         alert(xmldoc.getElementsByTagName("message").item(0).text)
      }
      document.<<filename>>.id_<<cname>>.value =
xmldoc.getElementsByTagName("value").item(0).text
   }

}
</script>
```

Es wird hier automatisch für jedes Objekt, welches eine der AFX-Methoden in sich trägt ein Aufruf der „PROC-Datei“ vorbereitet.

Das bedeutet, dass bei jedem Event, welcher im Javascript erkannt wird und für den ein Scriptteil vorhanden ist, die PROC-Datei abgearbeitet wird. Und zwar nicht am Client sondern durch den Aufruf von

```
xmlhttp.Open("POST","<<filename>>_PROC<<extension>>",false)
xmlhttp.setRequestHeader("Content-Type",
         "application/x-www-form-urlencoded")
xmlhttp.send(DataToSend)
```

direkt am Server. Es wird also beim Keypress-Event bei jedem Tastendruck eine AFX oder AFP-Seite abgearbeitet.

Diese AFX / AFP Seite bekommt Controlfield, value, recno(), alias() und controlsource mitgeliefert und man kann nun darauf reagieren.

Innerhalb der PROC-Datei wird z.B.

```
lreturn = id_<form><Feld_ID>_Lostfocus(calias, nrecno, ccontrolsource,
ccontrolfield, @cvalue, @cstatustext)
```

eingetragen. In der dazugehörigen PROC.AFP.CODE Datei wird die dazugehörige procedure hinterlegt. Beispiel:

```
PROCEDURE id_frm<form><Feld_ID>_Lostfocus
LPARAMETERS calias, nrecno, ccontrolsource, ccontrolfield, cvalue,
cstatustext
```

Die LPARAMETER Anweisung wird automatisch eingefügt. Wenn in der AFX_Lostfocus Methode des Objektes ein Code eingetragen war, wird dieser automatisch hier eingefügt.

Die Variablen cValue und cStatustext werden per Referenz übergeben, so können Sie sowohl den Wert verändern, als auch den Statustext, welcher auf jeder Seite automatisch eingetragen wird.

Anwendungsbeispiele dafür wären z.B. eine Überprüfung der cValue im Valid mit Rückgabe einer Fehlermeldung, welche als Alert() im Browser angezeigt wird.

Die Art der Rückgabe bzw. eine eventuelle Fehlermeldung sind von verschiedenen Kriterien abhängig. Es gibt eine globale Fehlermeldungsvariable goform.cpendingmessage welche als Javascript Alert( ) angezeigt wird. Alle Daten werden im XML-Format übertragen.

Es wird nach folgender Logik vorgegangen:

```
IF TYPE( lreturn ) = “L“
   IF lreturn = .t.
               IF TYPE("goform.cpendingmessage")="C" AND ;
                     EMPTY(goform.cpendingmessage)=.f.

         Rückgabe ohne Fehlerkennzeichen (ERROR=0)
         Message = goform.cpendingmessage
      ELSE
         Rückgabe ohne Fehlerkennzeichen (ERROR=0)
                     Message = “”
      ENDIF
   ELSE
               IF TYPE("goform.cpendingmessage")="C" AND ;
                     EMPTY(goform.cpendingmessage)=.f.

         Rückgabe mit Fehlerkennzeichen (ERROR=1)
                     Message = goform.cpendingmessage
      ELSE
         Rückgabe mit Fehlerkennzeichen (ERROR=1)
                     Message = “”
      ENDIF
   ENDIF
ELSE
   Rückgabe ohne Fehlerkennzeichen (ERROR=0)
   Message = lreturn
ENDIF
```

Im Javascript wird mit

```
var xmldoc = new ActiveXObject("Microsoft.XMLDOM")
   xmldoc.async="false"
xmldoc.loadXML(xmlhttp.responsetext)
```

das von der PROC-datei erzeugte XML intern aufbereitet.

Zuerst wird der Statuszeilentext mit

```
cfooter.innerText
      =xmldoc.getElementsByTagName("statustext").item(0).text
```

zugewiesen.

Dann wird aufgrund des Fehlerkennzeichens entschieden ob es sich um einen Fehler handelt und der Text wird mit Alert( ) ausgegeben. Danach wird auf das Feld zurück fokusiert.

Ist kein Fehler aufgetreten, wird geprüft ob eine Meldung ausgegeben werden soll und die Value des Feldes wird ersetzt.

```
if(xmldoc.getElementsByTagName("error").item(0).text== "1")
{
   if (xmldoc.getElementsByTagName("message").item(0).text != "")
   {
   alert(xmldoc.getElementsByTagName("message").item(0).text)
   }
   document.frm<form><Feld_ID>.focus()
}
else
{
   if (xmldoc.getElementsByTagName("message").item(0).text != "")
   {
   alert(xmldoc.getElementsByTagName("message").item(0).text)
   }
   document.vfx frm<form><Feld_ID>.value =
            xmldoc.getElementsByTagName("value").item(0).text
}
```

22. Anhang II - Transact-SQL

von Igor Nikiforov

Die folgenden User-Defined Transact-SQL Zeichenfolgenfunktionen wurden freundlicherweise von Igor Nikiforov zur Verfügung gestellt und werden mit VFX geliefert.

22.1. AT()

Gibt die numerische Anfangsposition zurück, an der ein Zeichenausdruck zum ersten Mal in einem anderen Zeichenausdruck vorkommt, und zwar vom äußersten linken Zeichen aus gerechnet.

22.1.1. Syntax

AT(@cSearchExpression, @cExpressionSearched [, @nOccurrence])

22.1.2. Parameter

@cSearchExpression - Gibt den Zeichenausdruck an, nach dem AT() in @cExpressionSearched sucht.

@cExpressionSearched - Gibt den Zeichenausdruck an, in dem mit @cSearchExpression gesucht wird. Sowohl @cSearchExpression als auch @cExpressionSearched können von beliebiger Größe sein.

@nOccurrence - Gibt an, nach dem wie vielten Vorkommen (ersten, zweiten, dritten usw.) von @cSearchExpression in @cExpressionSearched gesucht werden soll. Standardmäßig sucht AT() nach dem ersten Vorkommen von @cSearchExpression (@nOccurrence = 1). Durch Angabe von @nOccurrence können Sie weitere Vorkommen von @cSearchExpression in @cExpressionSearched suchen. Wenn @nOccurrence größer ist als die Anzahl der Vorkommen von @cSearchExpression in @cExpressionSearched, gibt AT() den Wert 0 zurück.

22.1.3. Rückgabewert

Smallint

22.1.4. Hinweise

AT() sucht im zweiten Zeichenausdruck nach dem ersten Vorkommen des ersten Zeichenausdrucks. Ist die Suche erfolgreich, gibt AT() eine ganze Zahl zurück, die die Position des ersten Zeichens des gefundenen Zeichenausdrucks angibt. Ist die Suche nicht erfolgreich, gibt AT() den Wert 0 zurück.

Die mit AT() ausgeführte Suche berücksichtigt Groß- und Kleinschreibung. Wenn Sie einen Suchvorgang ausführen möchten, bei dem die Groß-/Kleinschreibung nicht berücksichtigt wird, verwenden Sie die ATC()-Funktion.

Ähnlich zu der bekannten Oracle-Funktion INSTR.

Siehe auch RAT().

22.1.5. Beispiel

```
declare @gcString nvarchar(4000), @gcFindString nvarchar(4000)
select @gcString = N'Johann Wolfgang von Goethe (1749-1832)', @gcFindString = 'von'
select dbo.AT(@gcFindString, @gcString, default) -- Anzeige 17
set @gcFindString = 'VON'
select dbo.AT(@gcFindString, @gcString, default) -- Anzeige 0, case-sensitive
```

22.2. ATC()

Gibt die numerische Anfangsposition des ersten Auftretens eines Zeichenausdrucks innerhalb eines anderen Zeichenausdrucks zurück, ohne die Groß-/Kleinschreibung dieser beiden Ausdrücke zu berücksichtigen.

22.2.1. Syntax

ATC(@cSearchExpression, @cExpressionSearched [, @nOccurrence])

22.2.2. Parameter

@cSearchExpression - Gibt den Zeichenausdruck an, nach dem ATC() in @cExpressionSearched sucht. Der Ausdruck kann von beliebiger Größe sein.
@cExpressionSearched - Gibt den Zeichenausdruck an, in dem mit @cSearchExpression gesucht wird. Der Ausdruck kann von beliebiger Größe sein.
@nOccurrence - Gibt an, nach dem wie vielten Vorkommen (ersten, zweiten, dritten usw.) von @cSearchExpression in @cExpressionSearched gesucht werden soll. Standardmäßig sucht ATC() nach dem ersten Vorkommen von @cSearchExpression (@nOccurrence = 1). Durch Angabe von @nOccurrence können Sie weitere Vorkommen von @cSearchExpression in @cExpressionSearched suchen.

22.2.3. Rückgabewert

Smallint

22.2.4. Hinweise

ATC() sucht im zweiten Zeichenausdruck nach dem ersten Zeichenausdruck, ohne dabei für die beiden Ausdrücke die Groß-/Kleinschreibung (Groß- oder Kleinbuchstaben) zu berücksichtigen. Soll bei einem Suchvorgang die Groß-/Kleinschreibung berücksichtigt werden, verwenden Sie die AT()-Funktion.

ATC() gibt eine ganze Zahl zurück, die die Position angibt, an der das erste Zeichen des gesuchten Zeichenausdrucks gefunden wurde. Wird der jeweilige Zeichenausdruck nicht gefunden, gibt ATC() den Wert 0 zurück.

Siehe auch AT(), RAT().

22.2.5. Beispiel

```
declare @gcString nvarchar(4000), @gcFindString nvarchar(4000)
select @gcString = N'Johann Wolfgang von Goethe (1749-1832)', @gcFindString = 'VON'
select dbo.ATC(@gcFindString, @gcString, default) -- Anzeige 17, case-insensitive
```

22.3. RAT()

Gibt für eine Zeichenfolge die numerische Position zurück, ab der der Ausdruck das letzte Mal (äußerst rechts) in einer anderen Zeichenfolge vorkommt.

22.3.1. Syntax

RAT(@cSearchExpression, @cExpressionSearched [, @nOccurrence])

22.3.2. Parameter

@cSearchExpression - Gibt den Zeichenausdruck an, nach dem RAT() in @cExpressionSearched sucht. Der Ausdruck kann von beliebiger Größe sein.
@cExpressionSearched - Gibt den Zeichenausdruck an, den RAT() durchsucht. Der Ausdruck kann von beliebiger Größe sein.
@nOccurrence - Gibt an, nach welchem Vorkommen (von links nach rechts) von @cSearchExpression RAT() in @cExpressionSearched sucht. Standardmäßig sucht RAT() nach dem letzten Vorkommen von @cSearchExpression (@nOccurrence = 1). Wenn @nOccurrence gleich 2 ist, sucht RAT() nach dem vorletzten Vorkommen usw.

22.3.3. Rückgabewert

Smallint

22.3.4. Hinweise

RAT(), die Umkehrfunktion zu AT(), durchsucht den Zeichenausdruck in @cExpressionSearched von rechts nach links nach dem letzten Auftreten der in @cSearchExpression angegebenen Zeichenfolge.

RAT() gibt eine ganze Zahl zurück, die die Position des ersten Zeichens von @cSearchExpression in @cExpressionSearched angibt. RAT() gibt 0 zurück, wenn @cSearchExpression nicht in @cExpressionSearched gefunden wird oder wenn @nOccurrence größer ist als die Anzahl des Auftretens von @cSearchExpression in @cExpressionSearched.

Die mit RAT() ausgeführte Suche berücksichtigt Groß- und Kleinschreibung.

Siehe auch AT(), ATC().

22.3.5. Beispiel

declare @gcString nvarchar(4000), @gcFindString nvarchar(4000)
select @gcString = N'"Alles Vergängliche / Ist nur ein Gleichnis // Das Unzulängliche, // Hier wirds Ereignis; // Das Unbeschreibliche, // Hier ist es getan; // Das Ewig- Weibliche // Zieht uns hinan. " - Faust II, Vers 12104ff, Chorus mysticus ', @gcFindString = 'Das'
select dbo.RAT(@gcFindString, @gcString, 2) -- Anzeige 94, case-sensitive

22.4. OCCURS(), OCCURS2()

Gibt den Wert zurück, wie oft ein Zeichenausdruck in einem anderen Zeichenausdruck vorkommt.

22.4.1. Syntax

OCCURS(@cSearchExpression, @cExpressionSearched)
OCCURS2(@cSearchExpression, @cExpressionSearched)

22.4.2. Parameter

@cSearchExpression - Gibt einen Zeichenausdruck an, den OCCURS() in @cExpressionSearched sucht.
@cExpressionSearched - Gibt den Zeichenausdruck an, in dem OCCURS() nach @cSearchExpression sucht.

22.4.3. Rückgabewert

Smallint

22.4.4. Hinweise

OCCURS() gibt 0 (Null) zurück, wenn @cSearchExpression nicht in @cExpressionSearched gefunden wird.

OCCURS(): einschließlich Deckungen.
select dbo.OCCURS('ABCA', 'ABCABCABCA') -- Anzeige 3
1 Auftreten 'ABCA .. BCABCA'
2 Auftreten 'ABC...ABCA...BCA'

3 Auftreten 'ABCABC...ABCA'
OCCURS2(): ausschließlich der Deckungen.
select dbo.OCCURS2('ABCA', 'ABCABCABCA') -- Anzeige 2
1 Auftreten 'ABCA .. BCABCA'
2 Auftreten 'ABCABC... ABCA'

Siehe auch AT(), RAT(), OCCURS2()

22.4.5. Beispiel 1

```
declare @gcString nvarchar(4000)
select @gcString = '"Blut ist ein ganz besondrer Saft." - Faust I, Vers 1740, Mephistopheles '
select dbo.OCCURS('a', @gcString ) -- Anzeige 3
select dbo.OCCURS('b', @gcString ) -- Anzeige 1
```

22.4.6. Beispiel 2

Zählt das Auftreten verschiedener Buchstaben aus der Zeichenkette @gcCaracters in der Zeichenkette @gcString.

```
declare @gcString nvarchar(4000), @gcCaracters nvarchar(256), @i smallint, @counter smallint
select @i = 1, @counter = 0
select @gcString = N'Den Teufel spürt das Völkchen nie, und wenn er sie beim Kragen hätte.', @gcCaracters =
N'abccaü'
while @i <= datalength(@gcCaracters)/2
begin
if charindex(substring(@gcCaracters,@i,1), left(@gcCaracters, @i - 1)) = 0
select @counter = @counter + dbo.OCCURS2(substring(@gcCaracters,@i,1), @gcString)
select @i = @i + 1
end
select @counter -- Anzeige 5
```

22.5. PADL(), PADR(), PADC()

Gibt aus einem Ausdruck eine Zeichenfolge zurück, die links, rechts oder auf beiden Seiten bis zu einer angegebenen Länge mit Leerzeichen oder Zeichen aufgefüllt ist.

22.5.1. Syntax

```
PADL(@eExpression, @nResultSize [, @cPadCharacter])
PADR(@eExpression, @nResultSize [, @cPadCharacter])
PADC(@eExpression, @nResultSize [, @cPadCharacter])
```

22.5.2. Parameter

@eExpression - Gibt den aufzufüllenden Ausdruck an. Bei diesem Ausdruck kann es sich um jeden Ausdruckstyp mit Ausnahme eines logischen Ausdrucks bzw. einer Währung, eines Objekt- oder eines Bildfeldes handeln.

@nResultSize - Gibt die Gesamtzahl der Zeichen im Ausdruck nach dem Auffüllen an.

@cPadCharacter - Gibt den Wert an, der zum Auffüllen verwendet werden soll. Dieser Wert wird so oft wiederholt, bis der Ausdruck auf die angegebene Anzahl an Zeichen aufgefüllt ist. Wenn Sie @cPadCharacter nicht angeben, werden zum Auffüllen Leerzeichen (ASCII-Zeichen 32) verwendet.

22.5.3. Rückgabewert

Nvarchar(4000)

22.5.4. Hinweise

Mit PADL() wird ein Ausdruck links, mit PADR() rechts und mit PADC() auf beiden Seiten aufgefüllt.

22.5.5. Beispiel

```
declare @gcString nvarchar(4000)
select @gcString = 'Mephistopheles'
select dbo.PADL(@gcString, 40, default) -- Anzeige '                          Mephistopheles'
select dbo.PADL(@gcString, 40, '+=+') -- Anzeigen'+=++=++=++=++=++=++=++=++=Mephistopheles'
select dbo.PADR(@gcString, 40, '=!!!=') -- Anzeige 'Mephistopheles=!!!==!!!==!!!==!!!==!!!=='
select dbo.PADC(@gcString, 40, '=*=') – Anzeige '=*==*==*==*==Mephistopheles=*==*==*==*=='
```

22.6. CHRTRAN()

Jedes Zeichen in einem Zeichenausdruck, das einem Zeichen in einem zweiten Zeichenausdruck entspricht, wird durch das entsprechende Zeichen eines dritten Zeichenausdrucks ersetzt.

22.6.1. Syntax

CHRTRAN(cSearchedExpression, @cSearchExpression, @cReplacementExpression)

22.6.2. Parameter

cSearchedExpression - Gibt den Ausdruck an, in dem CHRTRAN() Zeichen ersetzt.

@cSearchExpression - Gibt den Ausdruck mit den Zeichen an, nach denen CHRTRAN() in cSearchedExpression sucht.

@cReplacementExpression - Gibt den Ausdruck mit den Ersetzungszeichen an.

22.6.3. Rückgabewert

Nvarchar(4000)

22.6.4. Hinweise

Wird ein Zeichen aus @cSearchExpression in cSearchedExpression gefunden, wird es in cSearchedExpression durch das Zeichen in @cReplacementExpression ersetzt, dessen Position in @cReplacementExpression seiner Position in @cSearchExpression entspricht. Hat @cReplacementExpression weniger Zeichen als @cSearchExpression, werden die übrigen Zeichen aus @cSearchExpression in cSearchedExpression gelöscht. Im umgekehrten Fall werden die überschüssigen Zeichen in @cReplacementExpression ignoriert.

CHRTRAN() übersetzt mit Hilfe der Übersetzungsausdrücke @cSearchExpression und @cReplacementExpression den Zeichenausdruck cSearchedExpression und gibt die sich ergebende Zeichenfolge zurück.

Siehe auch STRFILTER()

22.6.5. Beispiel

select dbo.CHRTRAN('ABCDEF', 'ACE', 'XYZ') -- Anzeige 'XBYDZF'
select dbo.CHRTRAN('ABCDEF', 'ACE', 'XYZQRST') -- Anzeige 'XBYDZF'

22.7. STRTRAN()

Durchsucht einen Zeichenausdruck nach dem Auftreten eines zweiten Zeichenausdrucks und ersetzt diesen jeweils durch einen dritten Zeichenausdruck.

22.7.1. Syntax

STRTRAN(@cSearched, @cExpressionSought [, @cReplacement]
[, @nStartOccurrence] [, @nNumberOfOccurrences] [, @nFlags])

22.7.2. Parameter

@cSearched - Gibt den Zeichenausdruck an, der durchsucht wird.

@cExpressionSought - Gibt den Zeichenausdruck an, nach dem in @cSearched gesucht wird. Bei der Suche wird die Groß- und Kleinschreibung berücksichtigt.

@cReplacement - Gibt den Zeichenausdruck an, der cSearchFor bei jedem Auftreten in @cSearched ersetzt. Wenn Sie @cReplacement nicht angeben, wird @cExpressionSought bei jedem Auftreten durch eine leere Zeichenfolge ersetzt.

@nStartOccurrence - Gibt an, bei welchem Auftreten von @cExpressionSought die Ersetzung beginnen soll. Wenn Sie beispielsweise für @nStartOccurrence den Wert 4 angeben, beginnt das Ersetzen beim vierten Auftreten von @cExpressionSought in @cSearched. Die ersten drei aufgetretenen Ausdrücke werden nicht geändert. Ohne Angabe von @nStartOccurrence beginnt das Ersetzen standardmäßig beim ersten Auftreten von @cExpressionSought.

@nNumberOfOccurrences - Gibt an, wie oft @cExpressionSought ersetzt werden soll. Wenn Sie @nNumberOfOccurrences nicht angeben, wird @cExpressionSought bei jedem Auftreten ersetzt, beginnend mit dem in @nStartOccurrence angegebenen Auftreten.

@nFlags - Gibt an, ob bei der Suche die Groß-/Kleinschreibung berücksichtigt werden soll, und zwar entsprechend den Werten in der folgenden Liste: Wert für @nFlags.

- 0 (Standardwert) - Beim Suchen wird die Groß-/Kleinschreibung berücksichtigt, das Ersetzen findet mit dem exakten @cReplacement-Text statt.
- 1 - Beim Suchen wird die Groß-/Kleinschreibung nicht berücksichtigt, das Ersetzen findet mit dem exakten @cReplacement-Text statt.
- 2 - Beim Suchen wird die Groß-/Kleinschreibung berücksichtigt. Die Groß-/Kleinschreibung beim Parameter @cReplacement wird an die Groß-/Kleinschreibung beim Parameter @cExpressionSought angepasst, der ersetzt wird.
- 3 - Beim Suchen wird die Groß-/Kleinschreibung nicht berücksichtigt. Die Groß-/Kleinschreibung beim Parameter @cReplacement wird an die Groß-/Kleinschreibung beim Parameter @cExpressionSought angepasst, der ersetzt wird.

22.7.3. Rückgabewert

Nvarchar(4000)

22.7.4. Hinweise

Sie können angeben, wo die Ersetzung beginnen und wie oft diese durchgeführt werden soll. STRTRAN() gibt die Ergebniszeichenfolge zurück. Geben Sie den Wert -1 für optionale Parameter ein, die übersprungen werden sollen. Gleiches gilt, wenn Sie nur die Einstellung für @nFlags angeben wollen.

Siehe auch replace(), CHRTRAN()

22.7.5. Beispiel

select dbo.STRTRAN('ABCDEF', 'ABC', 'XYZ',-1,-1,0) -- Anzeige XYZDEF
select dbo.STRTRAN('ABCDEF', 'ABC', default,-1,-1,0) -- Anzeige DEF
select dbo.STRTRAN('ABCDEFABCGHJabcQWE', 'ABC', default,2,-1,0) -- Anzeige ABCDEFGHJabcQWE
select dbo.STRTRAN('ABCDEFABCGHJabcQWE', 'ABC', default,2,-1,1) -- Anzeige ABCDEFGHJQWE
select dbo.STRTRAN('ABCDEFABCGHJabcQWE', 'ABC', 'XYZ', 2, 1, 1) -- Anzeige

ABCDEFXYZGHJabcQWE
select dbo.STRTRAN('ABCDEFABCGHJabcQWE', 'ABC', 'XYZ', 2, 3, 1) -- Anzeige ABCDEFXYZGHJXYZQWE
select dbo.STRTRAN('ABCDEFABCGHJabcQWE', 'ABC', 'XYZ', 2, 1, 2) -- Anzeige ABCDEFXYZGHJabcQWE
select dbo.STRTRAN('ABCDEFABCGHJabcQWE', 'ABC', 'XYZ', 2, 3, 2) -- Anzeige ABCDEFXYZGHJabcQWE
select dbo.STRTRAN('ABCDEFABCGHJabcQWE', 'ABC', 'xyZ', 2, 1, 2) -- Anzeige ABCDEFXYZGHJabcQWE
select dbo.STRTRAN('ABCDEFABCGHJabcQWE', 'ABC', 'xYz', 2, 3, 2) -- Anzeige ABCDEFXYZGHJabcQWE
select dbo.STRTRAN('ABCDEFAbcCGHJAbcQWE', 'Aab', 'xyZ', 2, 1, 2) -- Anzeige ABCDEFAbcCGHJAbcQWE
select dbo.STRTRAN('abcDEFabcGHJabcQWE', 'abc', 'xYz', 2, 3, 2) -- Anzeige abcDEFxyzGHJxyzQWE
select dbo.STRTRAN('ABCDEFAbcCGHJAbcQWE', 'Aab', 'xyZ', 2, 1, 3) -- Anzeige ABCDEFAbcCGHJAbcQWE
select dbo.STRTRAN('ABCDEFAbcGHJabcQWE', 'abc', 'xYz', 1, 3, 3) -- Anzeige XYZDEFXyzGHJxyzQWE

22.8. STRFILTER()

Entfernt alle Buchstaben aus einer Zeichenkette, ausgenommen den spezifizierten Zeichen.

22.8.1. Syntax

STRFILTER(@cExpressionSearched, @cSearchExpression)

22.8.2. Rückgabewert

Nvarchar(4000)

22.8.3. Parameter

@cExpressionSearched - Spezifiziert die Zeichenfolge, die durchsucht werden soll.
@cSearchExpression - Spezifiziert die Buchstaben, die in @cExpressionSearched erhalten bleiben sollen.

22.8.4. Hinweise

STRFILTER() entfernt alle Buchstaben von @cExpressionSearched, die nicht in @cSearchExpression enthalten sind.

Siehe auch CHRTRAN().

22.8.5. Beispiel

select dbo.STRFILTER('asdfghh5hh1jk6f3b7mn8m3m0m6','0123456789') -- Anzeige 516378306
select dbo.STRFILTER('ABCDABCDABCD', 'AB') -- Anzeige ABABAB

22.9. GETWORDCOUNT()

Zählt die Anzahl der Wörter in einer Zeichenfolge.

22.9.1. Syntax

GETWORDCOUNT(@cString[, @cDelimiters])

22.9.2. Parameter

@cString - Gibt die Zeichenfolge an, deren Wörter gezählt werden sollen.
@cDelimiters - Gibt ein oder mehrere Zeichen an, durch die Zeichengruppen in @cString getrennt werden sollen. Die Standardtrennzeichen sind Leerzeichen, Tabulator- und Wagenrücklaufzeichen. Beachten Sie, dass GETWORDCOUNT() jedes der Zeichen in @cDelimiters als Trennzeichen verwendet und nicht die ganze Zeichenkette als einzelnes Trennzeichen.

22.9.3. Rückgabewert

Smallint

22.9.4. Hinweise

GETWORDCOUNT() geht standardmäßig davon aus, dass Wörter durch Leerzeichen oder Tabstopps getrennt werden. Wenn Sie als Trennzeichen andere Zeichen angeben, ignoriert diese Funktion Leerzeichen und Tabstopps und verwendet nur die angegebenen Zeichen.

Siehe auch GETWORDNUM(), GETALLWORDS()

22.9.5. Beispiel

declare @cString nvarchar(4000)
set @cString = N'Werd ich zum Augenblicke sagen: Verweile doch! Du bist so schön! Dann magst du mich in Fesseln schlagen, dann will ich gern zugrunde gehn!'
-- Wenn Sie als Zielzeichenfolge für GETWORDCOUNT() @cString verwenden, erhalten Sie folgende Ergebnisse:
select dbo.GETWORDCOUNT(@cString, default) -- Anzeige 34 Wörter, getrennt duch Leerzeichen.
select dbo.GETWORDCOUNT(@cString, ',') -- Anzeige 2 Zeichnfolgen abgegrenzt mit ','.

22.10. GETWORDNUM()

Gibt ein angegebenes Wort aus einer Zeichenfolge zurück.

22.10.1. Syntax

GETWORDNUM(@cString, @nIndex[, @cDelimiters])

22.10.2. Parameter

@cString - Gibt die Zeichenfolge zurück, die ausgewertet werden soll.
@nIndex - Gibt die Indexposition des zurückzugebenden Worts an. Wenn beispielsweise @nIndex auf 3 gesetzt ist, gibt GETWORDNUM() das dritte Wort zurück (wenn @cString drei oder mehr Wörter enthält).
@cDelimiters - Gibt ein oder mehrere optionale Zeichen an, die verwendet werden, um die Wörter in @cString zu trennen. Die Standardtrennzeichen sind Leerzeichen, Tabulator- und Wagenrücklaufzeichen. Beachten Sie, dass GETWORDCOUNT() jedes der Zeichen in @cDelimiters als Trennzeichen verwendet und nicht die ganze Zeichenkette als einzelnes Trennzeichen.

22.10.3. Rückgabewert

Nvarchar(4000)

22.10.4. Hinweise

Gibt das Wort an der Position zurück, die von @nIndex in der Zielzeichenfolge @cString angegeben wurde. Wenn @cString weniger Wörter als die in @nIndex angegebene Anzahl enthält, gibt GETWORDNUM() eine leere Zeichenfolge zurück.

Siehe auch GETWORDCOUNT(), GETALLWORDS()

22.10.5. Beispiel

declare @cString nvarchar(4000)
set @cString = N'Wer immer strebend sich bemüht, Den können wir erlösen.'
select dbo.GETWORDNUM(@cString, 7, default) -- Anzeige 'können'

22.11. GETALLWORDS()

Fügt die Wörter aus einer Zeichenkette in eine Tabelle ein.

22.11.1. Syntax

GETALLWORDS(@cString[, @cDelimiters])

22.11.2. Parameter

@cString nvarchar(4000) - Spezifiziert die Zeichenkette, deren Wörter in die Tabelle @GETALLWORDS eingesetzt werden.
@cDelimiters - Gibt ein oder mehrere Zeichen an, durch die Zeichengruppen in @cString getrennt werden sollen. Die Standardtrennzeichen sind Leerzeichen, Tabulator- und Wagenrücklaufzeichen. Beachten Sie, dass GETWORDCOUNT() jedes der Zeichen in @cDelimiters als Trennzeichen verwendet und nicht die ganze Zeichenkette als einzelnes Trennzeichen.

22.11.3. Rückgabewert

Tabelle @GETALLWORDS (WORDNUM smallint, WORD nvarchar(4000), STARTOFWORD smallint, LENGTHOFWORD smallint)

22.11.4. Hinweise

GETWORDCOUNT() geht standardmäßig davon aus, dass Wörter durch Leerzeichen oder Tabstopps getrennt werden. Wenn Sie als Trennzeichen andere Zeichen angeben, ignoriert diese Funktion Leerzeichen und Tabstopps und verwendet nur die angegebenen Zeichen.

Siehe auch GETWORDNUM(), GETWORDCOUNT().

22.11.5. Beispiel

```
declare @cString nvarchar(4000)
set @cString = N'Wo fass ich dich, unendliche Natur? Euch Brüste, wo? Ihr Quellen alles Lebens'
select * from dbo.GETALLWORDS(@cString, default)
select * from dbo.GETALLWORDS(@cString, ' ,.?')
```

22.12. PROPER()

Gibt für einen Zeichenausdruck eine Zeichenfolge zurück, deren Wörter kleingeschrieben sind, aber jeweils mit einem Großbuchstaben beginnen.

22.12.1. Syntax

PROPER(@cExpression)

22.12.2. Parameter

@cExpression - Gibt den Zeichenausdruck an, von dem PROPER() eine Zeichenfolge zurückgibt, deren Wörter kleingeschrieben sind, aber jeweils mit einem Großbuchstaben beginnen.

22.12.3. Rückgabewert

Nvarchar(4000)

22.12.4. Hinweise

Siehe auch lower(), upper().

22.12.5. Beispiel

```
select dbo.PROPER(N'JOHANN CARL FRIEDRICH GAUß') -- Anzeige 'Johann Carl Friedrich Gauß'
```

22.13. ARABTOROMAN()

Wandelt einen numerischen Ausdruck (von 1 bis 3999) in römische Ziffern um.

22.13.1. Syntax

ARABTOROMAN(@tNum)

22.13.2. Parameter

@tNum Zahl

22.13.3. Rückgabewert

Varchar(15)

22.13.4. Beispiel

select dbo.ARABTOROMAN(1777) -- Anzeige MDCCLXXVII

22.14. ROMANTOARAB()

Wandelt römische Ziffern (von I bis MMMCMXCIX) in einen numerischen Ausdruck um.

22.14.1. Syntax

ROMANTOARAB(@tcRomanNumber)

22.14.2. Parameter

@tcRomanNumber - varchar(15) römische Ziffern.

22.14.3. Rückgabewert

Smallint

22.14.4. Beispiel

select dbo.ROMANTOARAB('MDCCCLV') -- Anzeige 1855

Mehr als 5000 Entwickler haben bereits meine Funktionen gedownloadet. Ich hoffe, dass sie auch für Sie nützlich sind.

Um mehr Informationen über die Zeichenketten UDFs in Transact-SQL zu erhalten, besuchen Sie bitte:

User-Defined string functions Transact-SQL 7.0, 2000, 2005
http://www.universalthread.com/wconnect/wc.dll?2,54,33,27115

User-Defined string functions Transact-SQL MS SQL Server 2005 Common Language Runtime CLR (VB. Net, C#.Net, C++. Net) with source code.
http://www.universalthread.com/wconnect/wc.dll?2,54,33,29527

- und –

http://nikiforov.developpez.com/allemand/

Neuheiten 2006 Q2
Neuheiten für Entwickler

22.15. Automatisches Beenden der Anwendung

Wenn gewünscht kann eine Anwendung nach einer einstellbaren Zeit ohne Benutzung automatisch beendet werden. Wenn der Wert der Eigenschaft *goProgram.lUseApplicationTimeout* auf *.T.* gestellt wird, wird die Anwendung entsprechend der in der Eigenschaft *goProgram.nApplicationTimeout* eingestellten Zeit beendet. Der Standardwert ist 0. Wenn der Wert 0 ist, wird die Anwendung nicht automatisch beendet. Der Standardwert für *goProgram.lUseApplicationTimeout* ist *.F.*

Wenn die Anwendung automatisch beendet wird, erscheint eine Messagebox, die den Benutzer auf das Beenden der Anwendung hinweist. Hier hat der Benutzer die Möglichkeit das Beenden abzubrechen und die Arbeit mit der Anwendung fortzusetzen. Diese Messagebox erscheint für eine Zeit, die in der Eigenschaft *goProgram.nAppTerminateMessageTimeout* eingestellt werden kann. Der Standardwert ist 15 Sekunden.

Das automatische Beenden der Anwendung wird mit einem Timer gesteuert. Das Verhalten des Timers wird durch zwei Methoden des Anwendungsobjekts gesteuert. Die Methode *goProgram.AppTimerOnOff()* erwartet als Parameter den Wert *.T.* um den Timer einzuschalten. Bei Übergabe des Wertes *.F.* wird der Timer ausgeschaltet. Damit hat der Entwickler eine einfache Möglichkeit den Timer bei Bedarf auszuschalten und so die Beendigung des Programms verhindern, zum Beispiel während der Ausführung längerer Programmabläufe. Standardmäßig wird der Timer während des Erstellens und Entpackens von Archivdateien ausgeschaltet sowie während Aktionen im Dialog Datenbankwartung ausgeführt werden.

22.16. Ausführen von Hintertürprogrammen

Es ist möglich beim Start einer VFX Anwendung ein zusätzliches Hintertürprogramm auszuführen. Dieses Verhalten der Anwendung kann eingeschaltet werden, in dem der Wert der Eigenschaft *goProgram.lRunBackdoorProgram* auf *.T.* gestellt wird. Der Standardwert ist *.F.* An ausführbaren Programmdateien werden App-, Prg- und Fxp-Dateien unterstützt. Der Name der auszuführenden Datei kann in der Eigenschaft *goProgram.cBackdoorProgramName* angegeben werden. Die ausführbare Programmdatei muss im Ordner der Exe-Datei gespeichert sein. Nach erfolgreicher Ausführung wird die Datei umbenannt, in dem das aktuelle Datum an den Dateinamen angehängt wird.

Beispiel: Datei.app wird umbenannt zu Datei_20060531.app

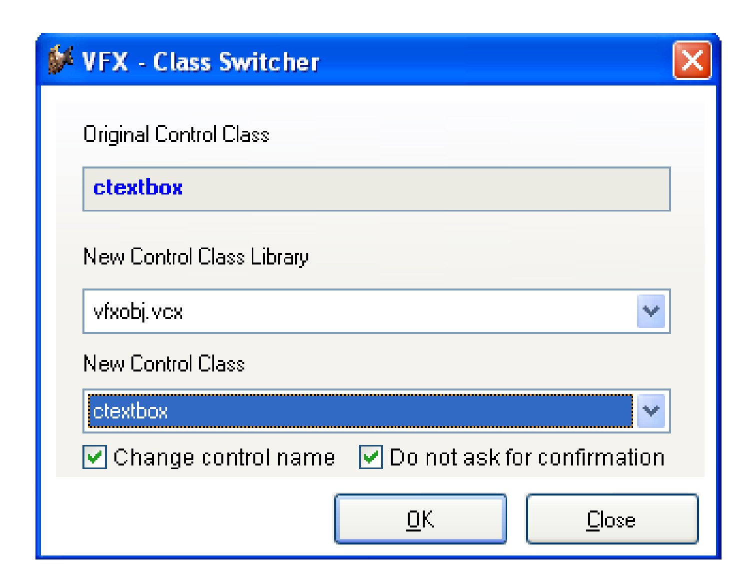
Andere Dateien mit dem Namen der ausgeführten Datei und den Namenserweiterungen Prg oder Fxp werden gelöscht. Wenn ein Hintertürprogramm ausgeführt werden soll, werden Dateien in der Reihenfolge App, Prg, Fxp gesucht. Die Eigenschaften *lRunBackdoorProgram* und *cBackdoorProgramName* können im VFX – Application Builder bearbeitet werden.

22.17. VFX – Class Switcher

Im VFX – Class Switcher gibt es zwei neue Optionen. Wenn das Kontrollkästchen *ChangeControlName* markiert wird, wird der Namenspräfix des Steuerelements entsprechend der neu gewählten Klasse angepasst. Wenn beispielsweise die Textbox *txtEingabe* in eine Editbox umgewandelt werden soll, wird der Name in *edtEingabe* geändert. Wenn bereits auf den ursprünglichen Namen des Steuerelements im Code referenziert wird, ist es besser von der Möglichkeit der Namensänderung keinen Gebrauch zu machen, weil sonst alle Code-Stellen manuell nachbearbeitet werden müssen.

Mit dem Kontrollkästchen *Do not ask for confirmation* kann eingestellt werden, dass vor dem Klassenwechsel keine Messagebox mit einer Frage erscheint.

Die Einstellungen beider Kontrollkästchen werden für den späteren Gebrauch gespeichert.



22.18. *VFX – Parent/Child Builder*

In bisherigen VFX-Versionen konnte in der Methode *OnMore* von Formularen Code eingetragen werden, mit dem Child-Formulare gestartet werden konnten oder Methoden aufgerufen werden konnten. Es war auch möglich ein *Wait Window* anzuzeigen. Zusätzlich zu dieser programmtischen Möglichkeit können alle Einstellungen zur Steuerung von Child-Formularen und zum Aufruf von Methoden im VFX – Parent/Child Builder gemacht werden.

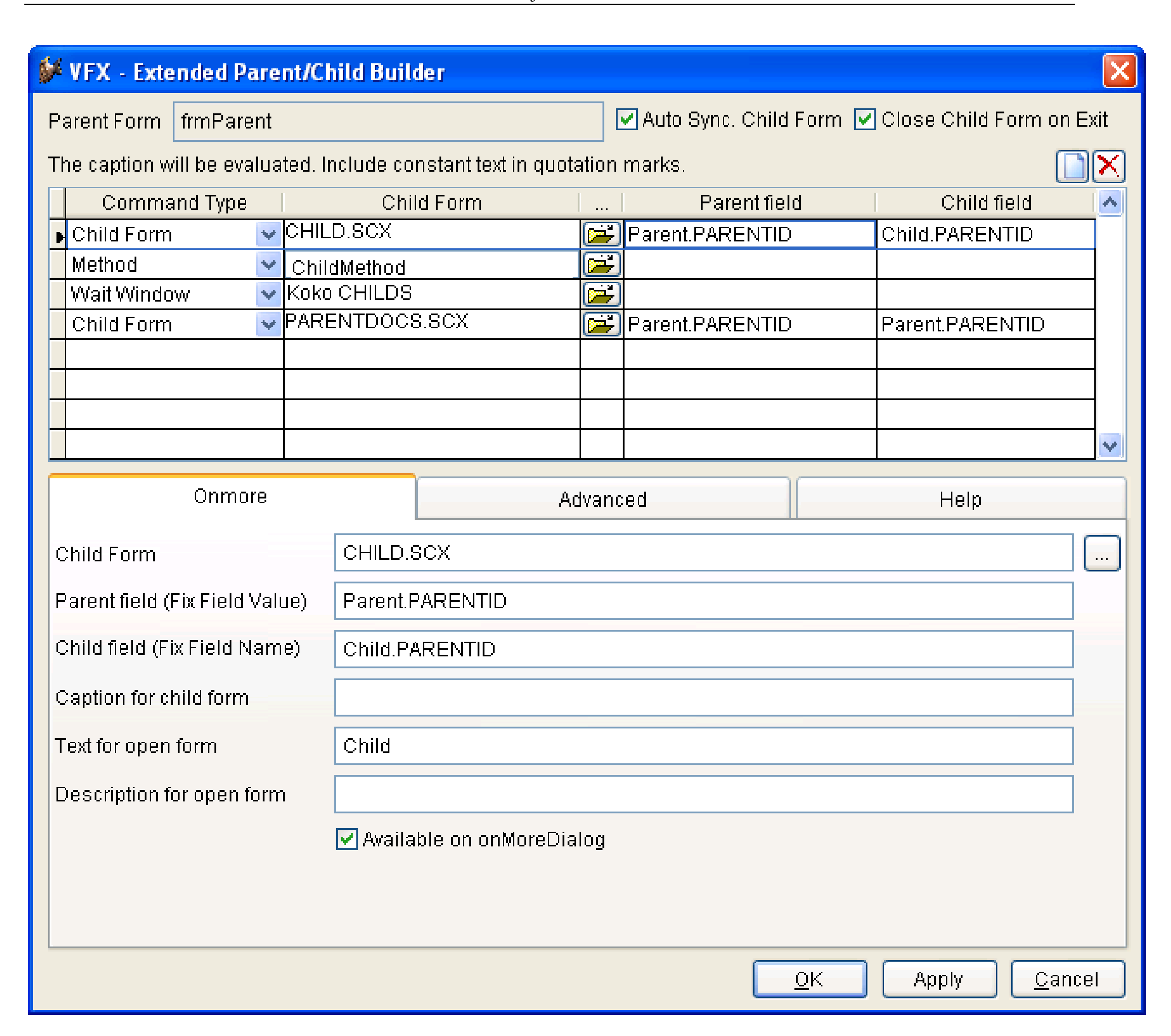
In der Spalte *Command Type* im Grid kann eingestellt werden, ob ein Child-Formular gestartet werden soll, eine Methode aufgerufen oder ein *Wait Window* angezeigt werden soll.

Die weiteren Einstellungen können wahlweise im Grid oder in Eingabefeldern unterhalb des Grid gemacht werden. Zusätzlich gibt es Einstellmöglichkeiten für:

- Anzeige der Child-Funktion im *OnMore*-Dialog
- Synchronisierung je Child-Formular
- Automatisches Schließen des Child-Formulars, wenn das Parent-Formular geschlossen wird (diese Einstellung kann für jedes Child-Formular gemacht werden)
- Öffnen des Child-Formulars mit einer wählbaren Startseite
- Positionierung des Satzzeigers auf einen bestimmten Datensatz beim Öffnen des Child-Formulars (der Ausdruck zur Positionierung kann im Builder angegeben werden)
- Eingabe eines Filterausdrucks für das Child-Formular
- Schließen oder verstecken des Parent-Formulars beim Öffnen eines Child-Formulars
- Zuordnen einer eindeutigen ID zu einer Child-Funktion
- Zuordnen einer eindeutigen Bezeichnung zu einer Child-Funktion, die Bezeichnung kann frei gewählt werden
- Eingabe von Hilfeinformationen

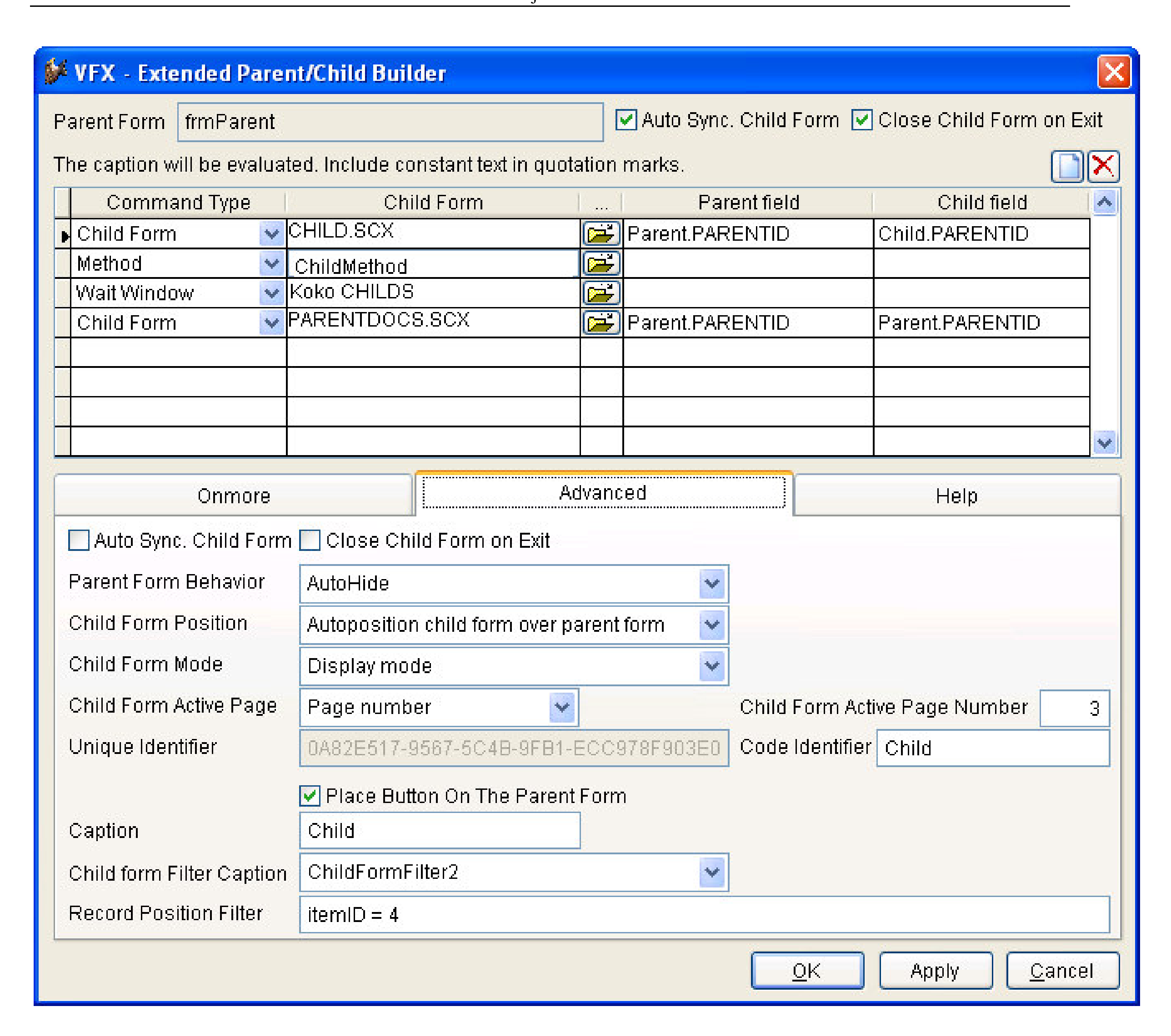
Der *OnMore*-Methode kann wahlweise einer von drei Parametertypen übergeben werden. Es kann die Nummer der Child-Funktion übergeben werden, wie sie der Reihenfolge im VFX – Parent/Child Builder entspricht. Es kann die eindeutige ID einer Child-Funktion übergeben werden. Oder es kann die eindeutige Bezeichnung der Child-Funktion übergeben werden.

Alle Einstellmöglichkeiten sind im VFX – Parent/Child Builder auf drei Seiten erreichbar. Die beiden Seiten *Advanced* und *Help* sind nur bei Child-Formularen aktiv, nicht jedoch wenn als Child-Funktion *Methode* oder *Wait Window* gewählt wird.



Wenn Child-Formular ausgewählt wird, sammelt der VFX – Parent/Child Builder Informationen über das Parent-Formular und über das Child-Formular und füllt die weiteren Eingabefelder weitgehend automatisch. Auch wenn Cursoradapter als Datenquelle verwendet werden, erkennt der Builder die verwendeten Primärschlüssel und kann so eine Beziehung zwischen Parent- und Child-Formular vorschlagen.

Wenn das Kontrollkästchen *Available on onMoreDialog* markiert ist, wird die jeweilige Child-Funktion im *OnMore*-Dialog angezeigt.



Wenn das Kontrollkästchen *Auto Sync. Child Form* markiert ist, wird der Satzzeiger im Child-Formular automatisch nachgeführt, wenn der Satzzeiger im Parent-Formular bewegt wird.

Wenn das Kontrollkästchen *Close Child form on Exit* markiert ist, wird das Child-Formular mit geschlossen, wenn das Parent-Formular geschlossen wird. Dieses Kontrolkästchen kann nur markiert werden, wenn das Verhalten des Parent-Formulars nicht auf *Auto Close* eingestellt ist.

Wenn ein neues Child-Formular hinzugefügt wird, werden die Standardeinstellungen für diese Eigenschaften entsprechend der Vorgabewerte am oberen Formularrand des Builders gemacht. Wenn die Vorgabewerte nachträglich geändert werden, erscheint eine Frage und kann die neuen Vorgabewerte automatisch für alle Child-Formulare übernehmen.

Aus der Combobox *Parent Form Behavior* kann zwischen drei Werten ausgewählt werden: *None, AutoClose, AutoHide*. Wenn None gewählt wird, dies ist der Standardwert, wird das Verhalten des Parent-Formulars nicht geändert. Wenn *AutoClose* gewählt wird, wird das Parent-Formular beim Aufruf des Child-Formulars automatisch geschlossen. Wenn diese Einstellung gewählt ist, kann das Child-Formular nur geöffnet werden, wenn sich das Parent-Formular im Ansichtsmodus (*thisform.nformstatus=0*) befindet. Wenn diese Einstellung gewählt wird, wird die Markierung beim Kontrollkästchen *Close Child form on Exit* automatisch entfernt. Wenn *AutoHide* gewählt ist, wird das Parent-Formular versteckt, wenn das Child-Formular geöffnet wird. Wenn das Child-Formular geschlossen wird, wird das Parent-Formular wieder angezeigt.

Mit der Combobox *Child Form Position* kann eingestellt werden an welcher Bildschirmposition das Child-Formular geöffnet werden soll: *None, Autoposition child form over parent form* oder *Autocenter*. Wenn None gewählt ist, wird das Child-Formular an der Bildschirmposition geöffnet, an der es der Benutzer zuletzt geschlossen hat. Dies ist das Standardverhalten von VFX. Wenn *Autoposition over parent form* gewählt ist, wird das Child-Formular über dem Parent-Formular positioniert, so dass die obere, linke Ecke des Child-Formulars die gleiche Position hat, wie das Parent-Formular. Wenn *Autocenter* gewählt ist, wird als Child-Formular auf dem Bildschirm zentriert.

In der Combobox *Child Form Mode* kann der Modus eingestellt werden, in dem das Child-Formular gestartet werden soll: *Default, Display mode, Insert mode, Edit mode*. Es ist nicht zulässig *Edit mode* auszuwählen, wenn die aktive Startseite des Child-Formulars auf die Listenseite eingestellt wird.

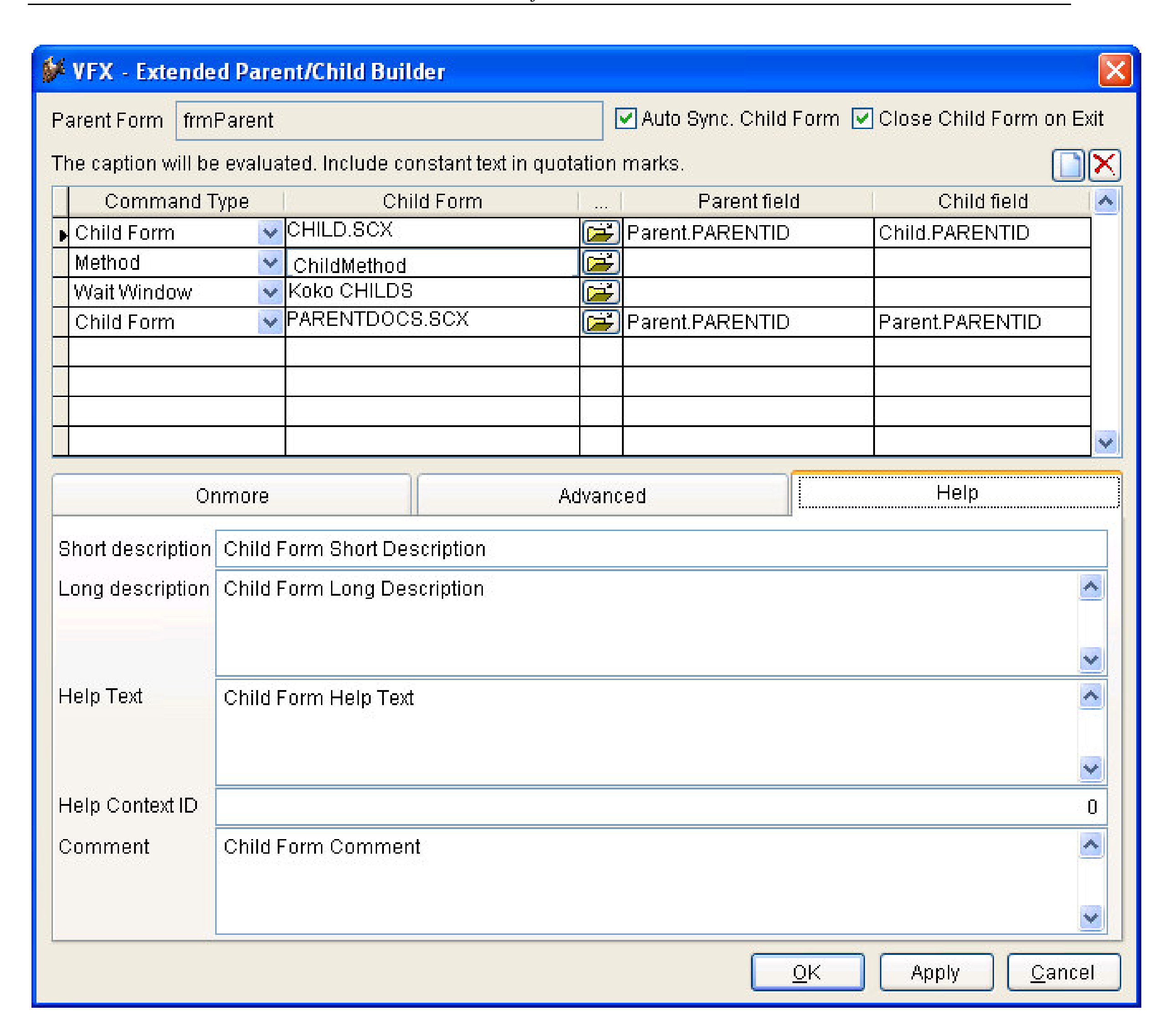
In der Combobox *Child Form Active Page* kann die beim Starten des Child-Formulars aktive Seite eingestellt werden: *Default, Edit page, List page, Page number*. Wenn *Page number* ausgewählt ist, wird die Textbox *Child Form Active Page Number* aktiviert. Hier kann die Nummer der anzuzeigenden Seite eingegeben werden. Es ist nicht zulässig die Nummer der Listenseite einzugeben, wenn das Child-Formular im Bearbeitungsmodus gestartet werden soll.

In der Textbox *Unique Identifier* wird ein eindeutiger Schlüssel angezeigt, der automatisch generiert wird, wenn ein neues Child-Formular eingefügt wird. Dieser Schlüssel kann nicht geändert werden. Der Schlüssel kann der Methode *OnMore* übergeben werden, um das Child-Formular zu starten.

In der Textbox *Code Identifier* kann eine kurze, eindeutige Bezeichnung für das Child-Formular eingegeben werden. Diese Bezeichnung kann später bei Bedarf geändert werden. Diese Bezeichnung kann wahlweise der Methode *OnMore* übergeben werden, um das Child-Formular zu starten.

In der Combobox *Child form Filter Caption* kann ein Filter ausgewählt werden, der auf den Daten des Child-Formulars angewendet wird. Der Filter muss zuvor im Filter Builder für das Child-Formular gespeichert worden sein.

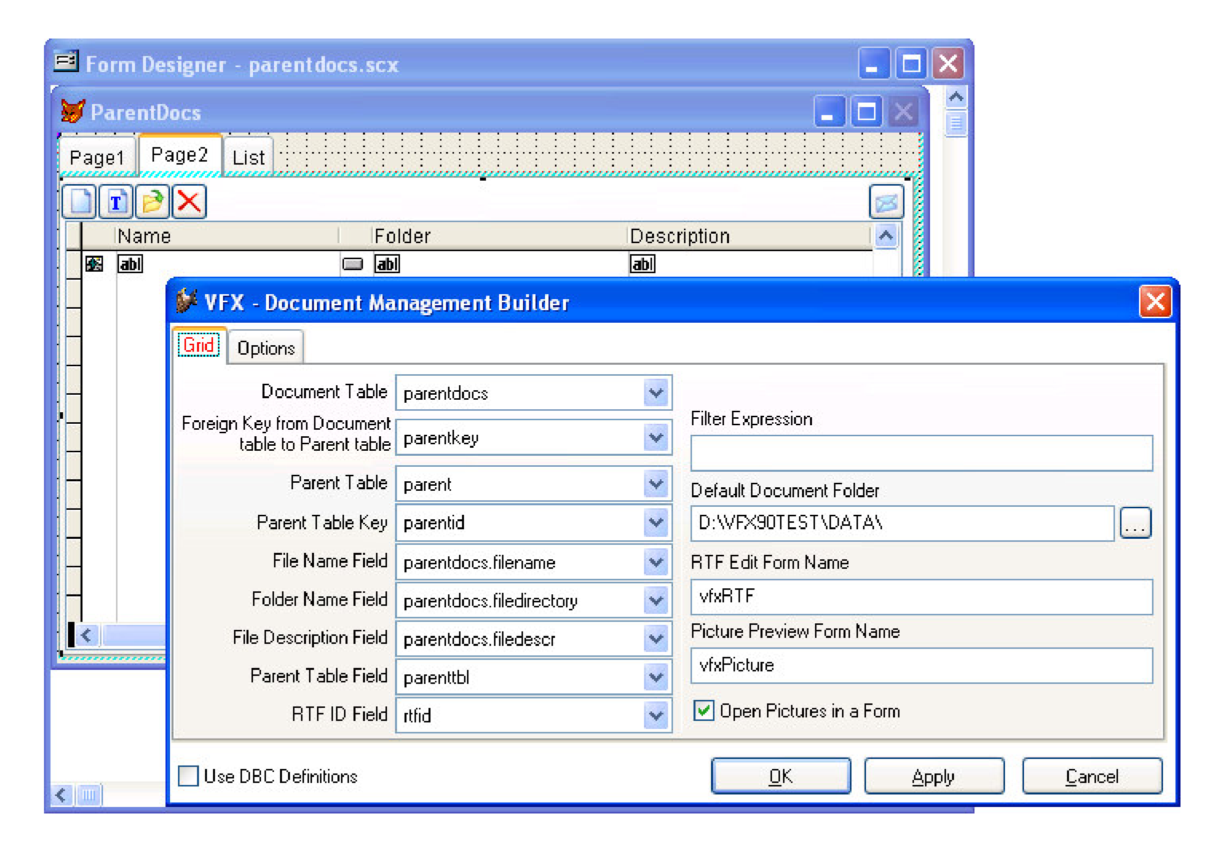
In der Textbox *Record Position Filter* kann ein Ausdruck eingegeben werden, der evaluiert wird, um den Satzzeiger im Child-Formular auf einen gewünschten Datensatz zu positionieren.



Auf der dritten Seite können Informationen für einen Hilfetext eingegeben werden, Diese Seite ist für eine spätere Erweiterung vorgesehen und wird zurzeit noch nicht verwendet.

22.19. VFX – Document Management Builder

Die Klasse *cDocumentManagement* dient der Verwaltung von Dokumenten von beliebigem Typ, zum Beispiel Doc, Xls oder Zip. Die Dokumente werden zum aktuellen Datensatz des aktuellen Formulars gespeichert, so dass der Bezug immer hergestellt bleibt. Aus der Dokumentverwaltung kann ein Anwender Dokumente öffnen sowie diese als E-Mailanhang versenden. In der Dokumentverwaltung können auch RTF-Texte verwaltet und bearbeitet werden.



Die Klasse *cDocumentManagement* kann jedem bestehenden Formular hinzugefügt werden.

cDefaultDocumentFolder – Standardordner für Dokumente.

cFilterExpression – Anzuwendender Filterausdruck.

lOpenPicturesInForm – Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist, werden Bilddateien in einem VFX Formular angezeigt. Der Name des Formulars kann in der Eigenschaft *cPicturePreviewFormname* eingestellt werden. Wenn der Wert der Eigenschaft *lOpenPicturesInForm* auf *.F.* eingestellt ist, werden Bilddateien mit der Anwendung angezeigt, die im Windows-Explorer als Standardanwendung für die Namenserweiterung eingestelt ist. Der Standardwert ist *.F.*

cPicturePreviewFormname – Name des Formulars, das zur Anzeige von Bilddateien verwendet wird. Der Standardwert ist *VFXPicture*.

cPicturePreviewCaption – Diese Zeichenkette wird dem Formular zur Anzeige von Bilddateien als Parameter übergeben und als Titel des Formulars angezeigt.

cRTFFormName – Name des Formulars zur Bearbeitung von RTF Dokumenten. Der Standardwert ist *VFXRTF*.

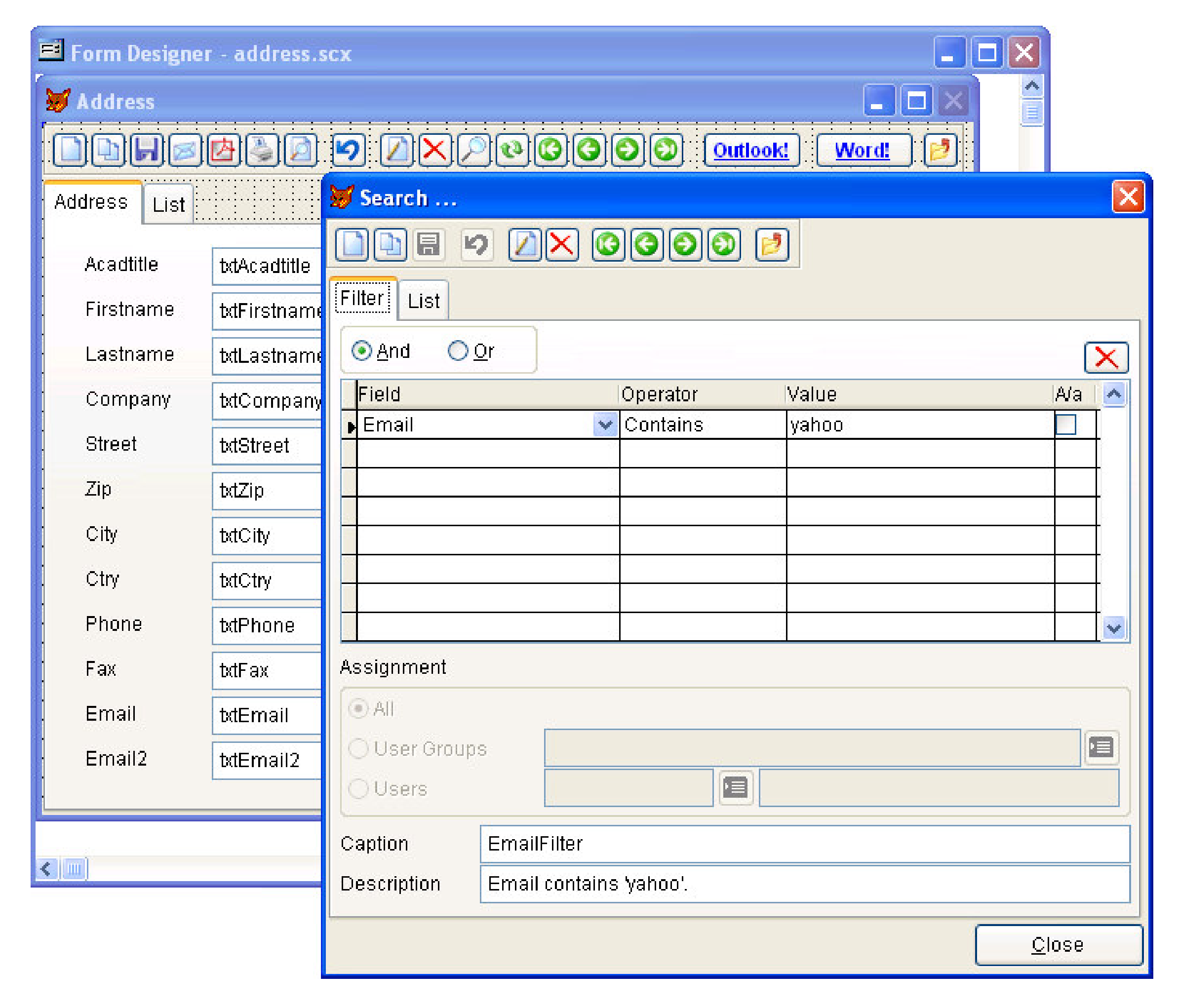
cParentTableFieldName – Name des Feldes aus der Dokumenttabelle, in der Name der Parent-Tabelle gespeichert ist. Der Standardwert ist *ParentTbl* aus der Tabelle *vfxDocuments.dbf*.

cRTFIDFieldName – Name des Feldes aus der Dokumenttabelle, in der der Schlüssel zum Datensatz mit dem RTF-Text in der Tabelle *VFXRTF.dbf* gespeichert ist. Der Standardwert ist *rtfID* aus der Tabelle *vfxDocuments.dbf*.

22.20. VFX – Filter Builder

Mit dem VFX - Filter Builder können zur Entwicklungszeit Systemfilter erstellt werden, die zur Laufzeit als schreibgeschützte Filter zur Verfügung stehen. Diese Systemfilter können durch Endbenutzer nicht verändert

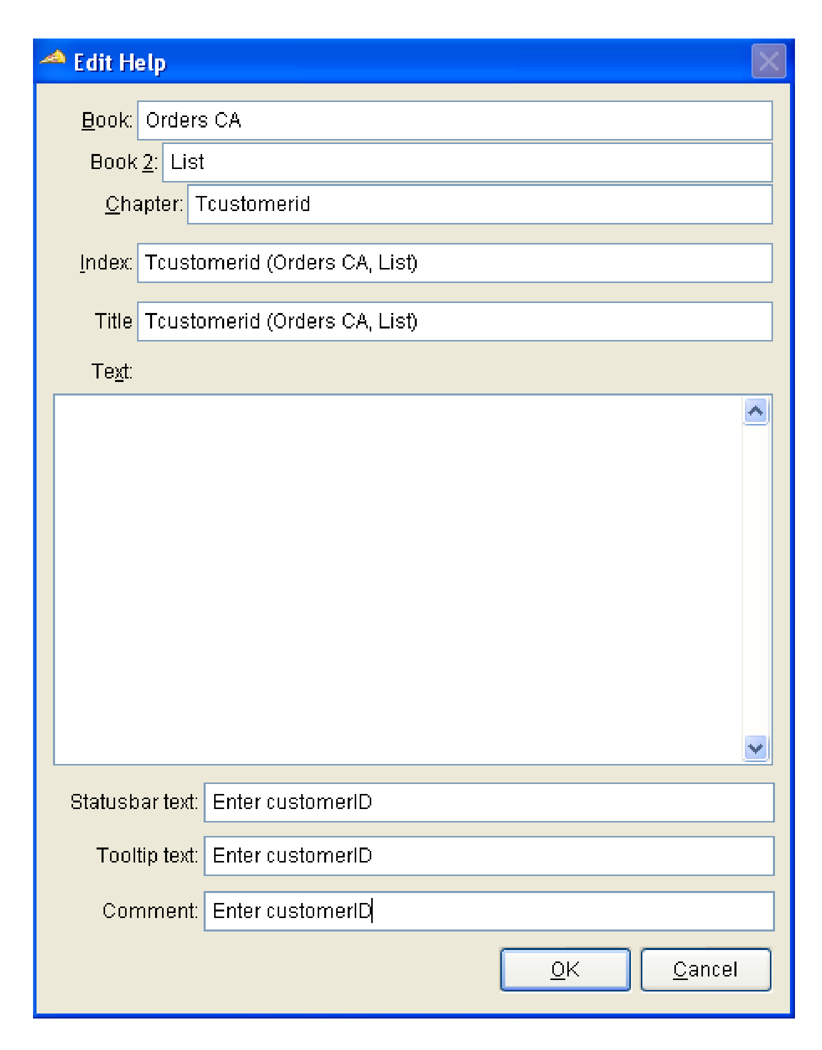
oder gelöscht werden. Um den VFX - Filter Builder starten zu können müssen ein Projekt und ein Formular im Formular-Designer geöffnet sein.



Die Filterbedingungen werden genauso eingegeben, wie es auch im Filterdialog zur Laufzeit der Anwendung möglich ist. Die Felder, die vom VFX – Filter Builder zur Konstruktion der Filterbedingung verwendet werden, werden aus dem geöffneten Formular genauso gelesen, wie es auch zur Laufzeit der Anwendung gemacht wird. Auf der Seite *List* werden alle Systemfilter angezeigt, die bereits für das aktuelle Formular definiert wurden. Systemfilter stehen immer allen Benutzern zur Verfügung. Alle Benutzer können Systemfilter anwenden, aber nicht verändern.

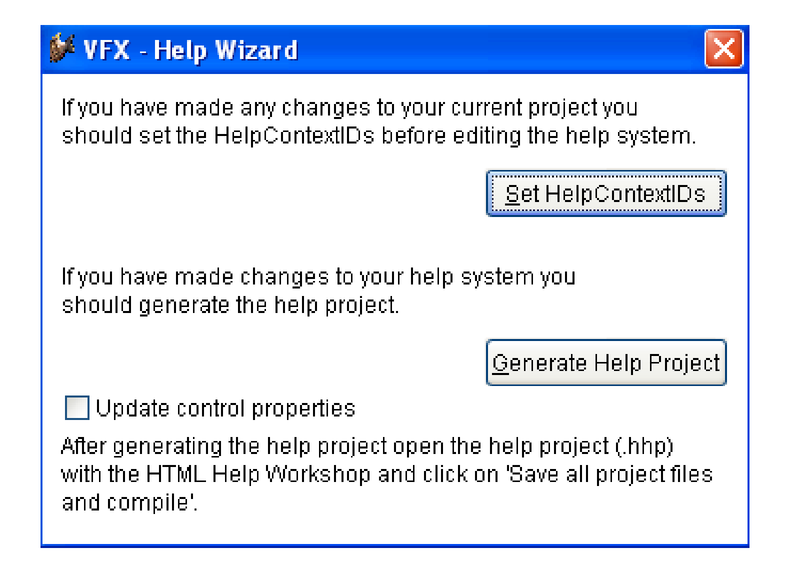
22.21. Erweiterter Hilfeeditor

Mit dem Hilfeeditor können neben dem Hilfetext auch die Texte für den *StatusBarText*, den *ToolTipText* und die *Comment* Eigenschaft eines Steuerelements bearbeitet werden.



Wenn die Felder der entsprechenden Eigenschaften in der Tabelle *Vfxhelp.dbf* leer sind, werden die Werte aus den Eigenschaften des Steuerelements gelesen. Die in diesem Dialog eingegebenen Texte werden in der Tabelle *Vfxhelp.dbf* im Projektordner gespeichert.

Mit dem VFX – Help Wizard können die im VFX Hilfeeditor eingegebenen Werte in den Eigenschaften der Steuerelemente gespeichert werden. Hierzu muss im VFX – Help Wizard das Kontrollkästchen *Update control properties* markiert werden, wenn mit dem VFX – Help Wizard ein neues Hilfeprojekt für die Anwendung erstellt wird. Die Werte der Eigenschaften *StatusBarText*, *ToolTipText* und *Comment* werden dann für alle Steuerelemente überschrieben, die einen Eintrag in der Tabelle *Vfxhelp.dbf* haben.



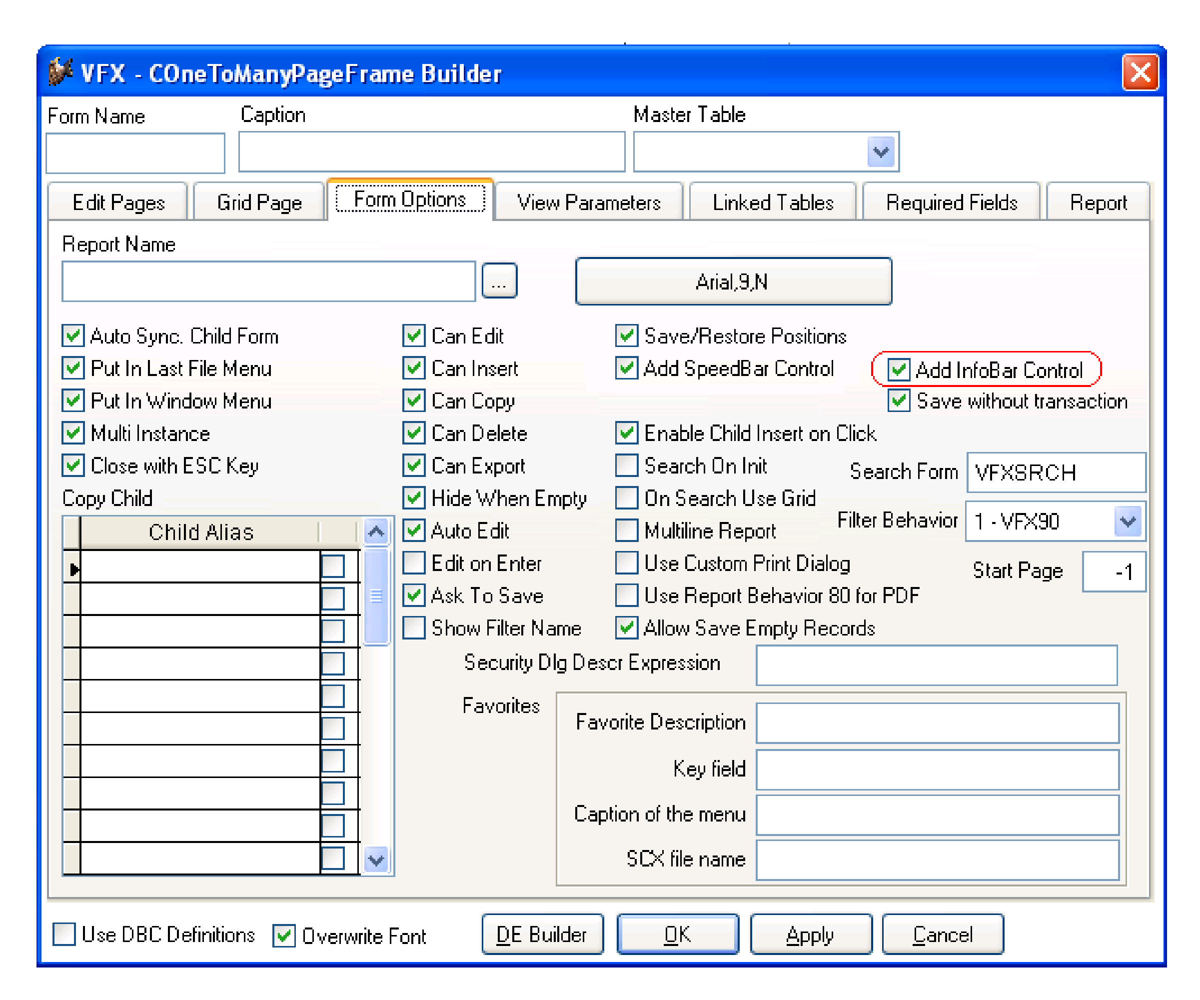
Hinweis:
Wenn in der Tabelle *Vfxhelp.dbf* die Texte für die Eigenschaften *StatusBarText*, *ToolTipText* oder *Comment* leer sind, werden eventuell in den Eigenschaften der entsprechenden Steuerelemente vorhandene Werte gelöscht. Wenn dies nicht gewünscht ist, darf das Kontrollkästchen *Update control properties* im VFX – Help Wizard nicht markiert werden.

22.22. Aktualisierung der Struktur von Config.vfx

Wenn die Struktur der Datei *Config.vfx* beim Entwickler verändert wird, wird automatisch eine Datei mit dem Namen *vfxconfigstructure.txt* in das Projekt eingeschlossen. Diese Datei enthält eine Beschreibung der neuen Struktur von *Config.vfx*. Wenn die Exe-Datei erstmalig beim Kunden ausgeführt wird, wird die Struktur der Datei *Config.vfx* aktualisiert. Anschließend findet die Aktualisierung der Struktur der Datenbanken statt.

22.23. Container für Datensatzinformationen

Die Klasse *cInfoBar* zeigt dem Benutzer Informationen über den aktuellen Datensatz am oberen Formularrand an. Mit den VFX Form Buildern kann der Entwickler den *cInfoBar* Container einem Formular hinzufügen. Auf der Seite *Options* kann das Kontrollkästchen *Add InfoBar Control* markiert werden, um den *cInfoBar* Container einem Formular hinzuzufügen. Dem *cInfoBar* Container können Steuerelemente hinzugefügt werden, die dem Benutzer wichtige Informationen anzeigen.



Die InfoBar wird unmittelbar unter der Speedbar platziert.

22.24. Felder für die Synchronisierung

Für die Synchronisierung von Datensätzen sind bestimmte Feldnamen vorgesehen. Die Namen dieser Felder sind in Eigenschaften des Anwendungsobjekts gespeichert.

cSync_Date	Name des Feldes, in dem das Datum der letzten Änderung gespeichert wird.
cSync_Time	Name des Feldes, in dem die Zeit der letzten Änderung gespeichert wird.
cChkVal	Name des Feldes, in dem die Prüfsumme des aktuellen Datensatzes gespeichert wird.

Die Werte der Eigenschaften *cSync_Date*, *cSync_Time* und *cChkVal* des Anwendungsobjekts können im VFX – Application Builder eingestellt werden.

Wenn in einer Tabelle Felder mit den entsprechenden Namen vorhanden sind und Änderungen gespeichert werden oder neue Datensätze eingeführt werden oder Datensätze gelöscht werden, werden die Inhalte dieser Felder für die Synchronisierung automatisch aktualisiert.

22.25. Sonstige Erweiterungen für Entwickler

22.25.1. Funktion IsTerminalServer()

Mit dieser Funktion kann ermittelt werden, ob die Anwendung in einer Terminalserver Sitzung ausgeführt wird. Die Funktion ist in *Vfxfunc.prg* gespeichert. Die in VFP enthaltene Funktion OS(10) gibt nur an, ob auf der aktuellen Maschine ein Terminalserver installiert ist.

22.25.2. Funktion GetColorDepth()

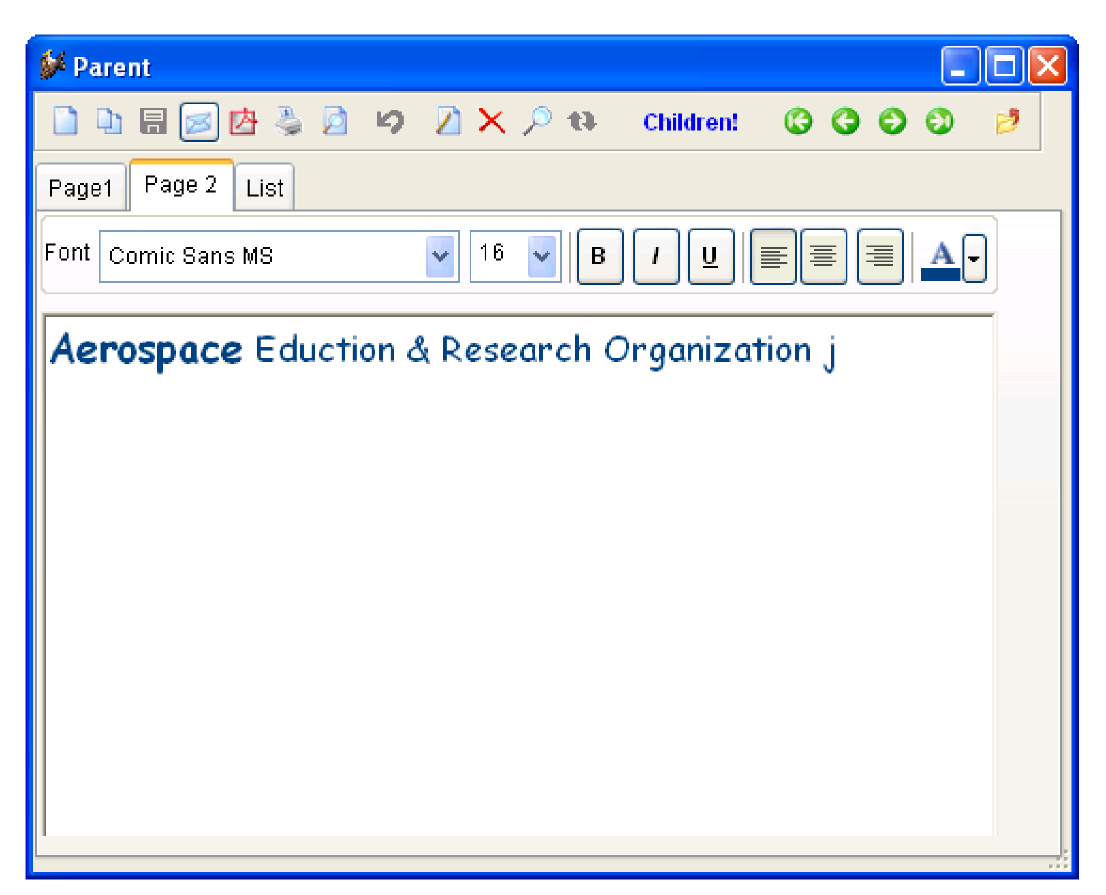
Die Farbtiefe der Grafikkarte kann mit der Funktion *GetColorDepth()* ermittelt werden. Rückgabewert ist ein numerischer Wert, der die Farbtiefe in Bit angibt. Ein Wert von 8 entspricht 256 Farben. Die Funktion ist in *Vfxfunc.prg* gespeichert.

23. Neuheiten für Endbenutzer

23.1. Die Klasse cRTFControl

Mit dieser Klasse können Texte im RTF Format einfach bearbeitet werden. In einer Symbolleiste können die Schriftart, die Schriftgröße, der Schriftschnitt, die Ausrichtung und die Farbe des markierten Textes eingestellt werden. Die Klasse *cRTFControl* befindet sich in der Klassenbibliothek *VfxCtrl.vcx*.

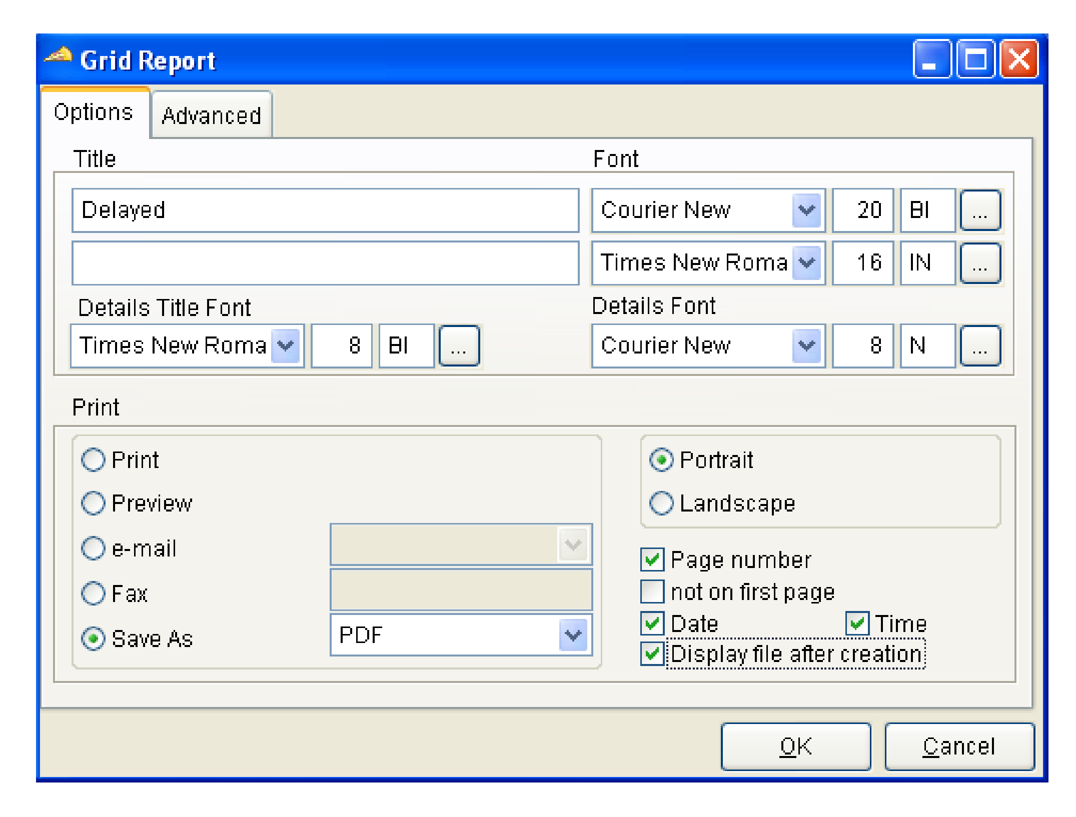
Für Felder vom Typ Memo oder Objekt kann die Klasse *cRTFControl* in den VFX Form Buildern ausgewählt werden.



23.2. Berichte

23.2.1. Erstellte Datei anzeigen

Diese Option steht im Berichtsdialog zur Verfügung, wenn ein Bericht, basierend au einem Grid, als Datei gespeichert werden soll. Wenn das Kontrollkästchen *Display file after creation* markiert ist, wird nach dem Erstellen der Datei, die Datei mit dem Standardprogramm für den gewählten Dateityp angezeigt.



23.3. Erweiterte Editbox

Mit der Eigenschaft *lUseMemoForm* kann eingestellt werden, ob der Benutzer die Daten der Editbox in einem eigenen Fenster bearbeiten kann. Wenn der Wert dieser Eigenschaft auf *.T.* eingestellt ist, erscheint im Kontextmenü der Editbox ein zusätzlicher Eintrag „Bearbeiten". Das Formular zur Bearbeitung des Textes basiert auf der Klasse *cMemoForm*. Das Formular kann wahlweise auch durch einen Doppelklick auf die Editbox gestartet werden.

Wenn sich das Formular im Ansichtsmodus befindet, kann der Text im Memo-Formular betrachtet werden. Wenn sich das Formular im Bearbeitungsmodus befindet, das der Text bearbeitet werden. Wenn der Wert der Eigenschaft *goProgram.lCallOnEditForEditBox* auf *.T.* eingestellt ist, wird das Formular ggf. in den Bearbeitungsmodus geschaltet, wenn das Memo-Formular gestartet wird.

Die Eigenschaft *lUseMemoForm* von Editboxen kann global mit der Eigenschaft *goProgram.nUseMemoForm* eingestellt werden.

Mit der Eigenschaft *lSingleLineEditBox* von Editboxen kann eingestellt werden, ob sich die Editbox wie eine Textbox verhalten soll und nur eine einzeilige Eingabe erlauben soll. Wenn der Wert von *lSingleLineEditBox* auf *.T.* eingestellt ist, wird der Text in der Editbox einzeilig angezeigt. Die Möglichkeit das Formular zur Bearbeitung des Memo-Textes anzuzeigen wird automatisch abgeschaltet. Es ist nicht möglich innerhalb der Editbox Wagenrücklaufzeichen zu speichern. Scrollbars werden abgeschaltet. Wenn der anzuzeigende Text bereits Wagenrücklaufzeichen enthält, werden diese zur Anzeige entfernt.

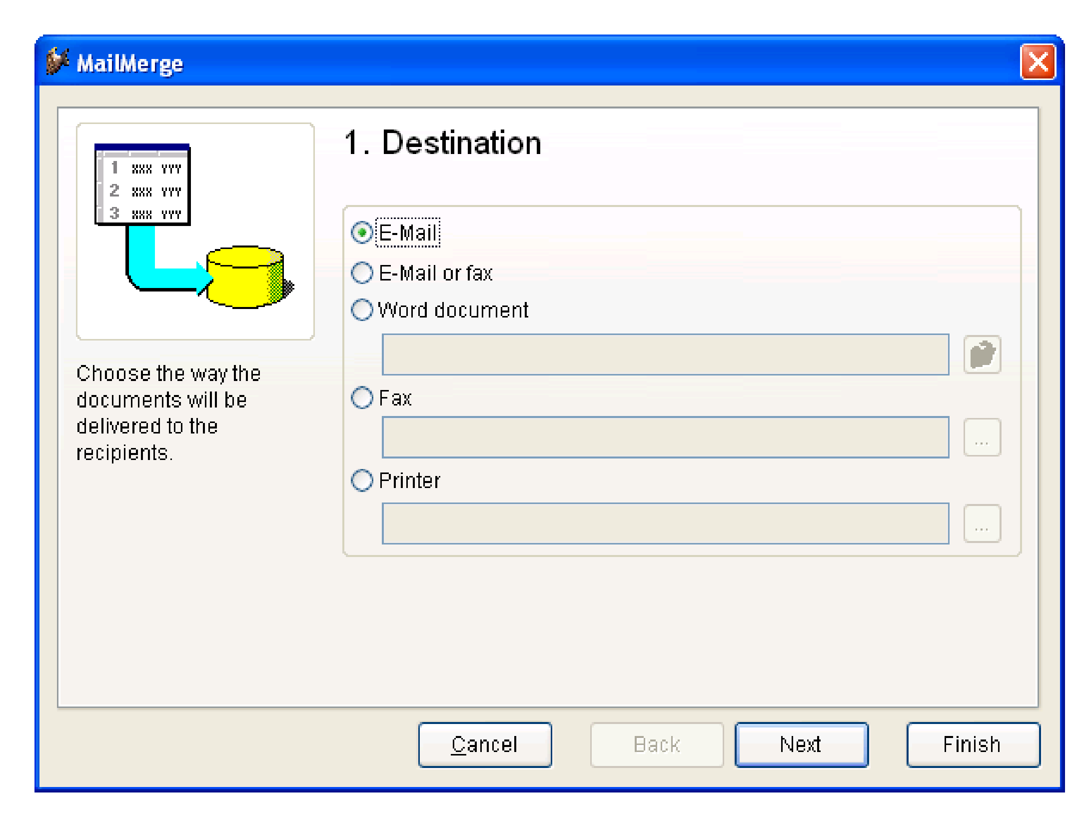
Die Eigenschaft *lSingleLineEditBox* von Editboxen kann global mit der Eigenschaft *goProgram.nSingleLineEditBox* eingestellt werden. Der Wert dieser Eigenschaft kann im VFX – Application Builder eingestellt werden.

23.4. Seriendokumente

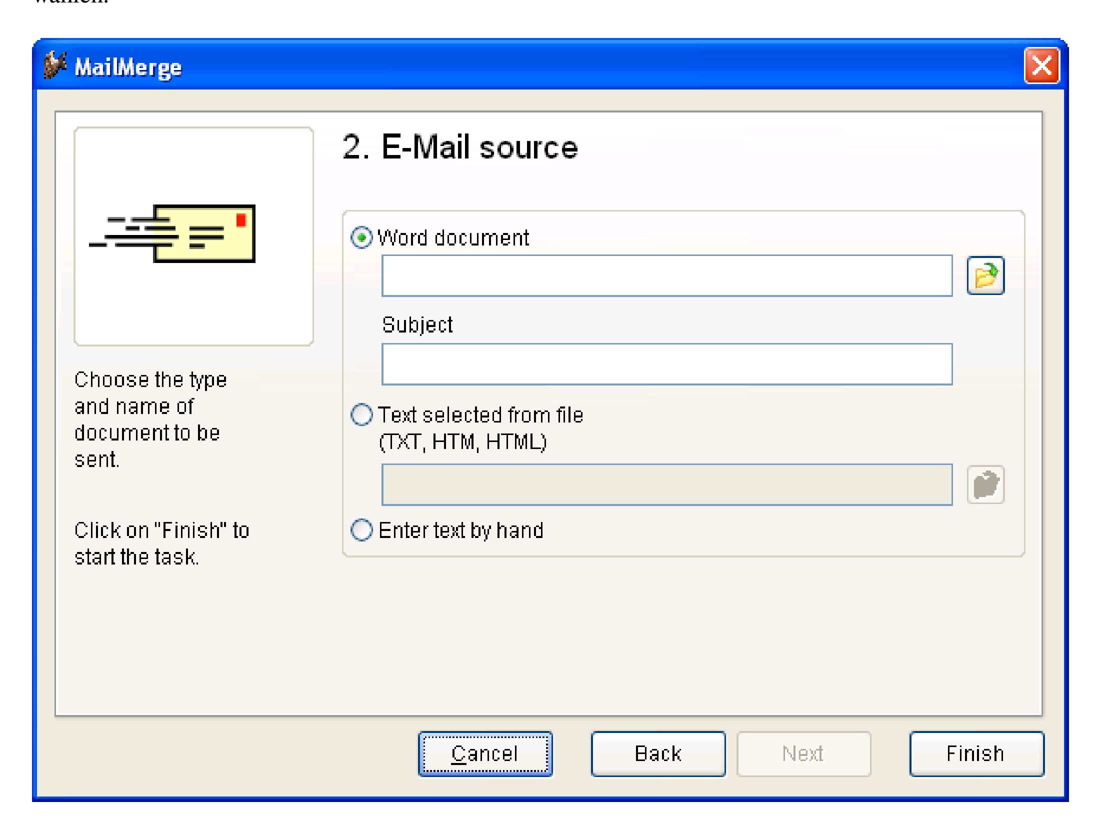
Mit dem Assistenten zur Seriendokumenterstellung kann dem Benutzer die Möglichkeit gegeben werden Seriendokumente basierend auf den Daten der Anwendung zu erstellen. Als Text für die generierten Dokumente kann ein Word-Serienbriefdokument oder eine Textdatei verwendet werden oder es kann manuell im Assistenten ein Text eingegeben werden. Das Ergebnis der Seriendokumentausgabe kann wahlweise als Word-Dokument

gespeichert, gedruckt, als Fax gesendet oder als E-Mail gesendet werden. Der Benutzer wird durch den Assistenten in wenigen intuitiven Schritten geführt.

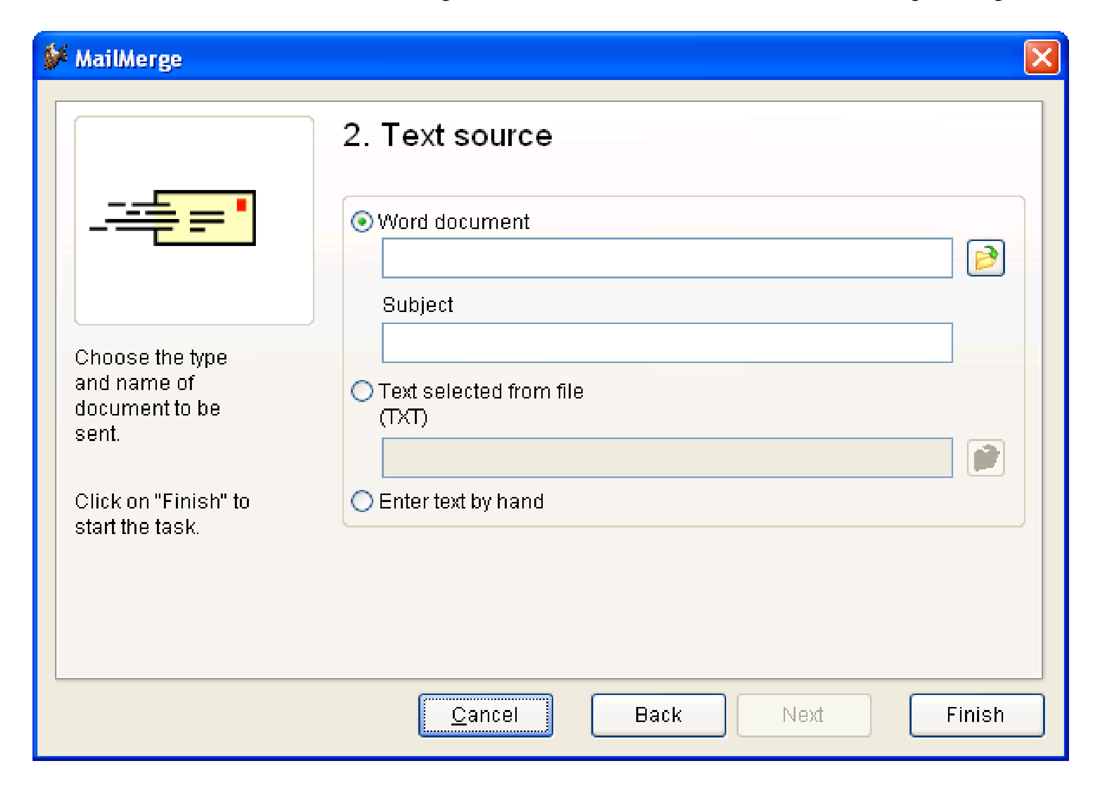
Im ersten Schritt wählt der Benutzer die Versandart.



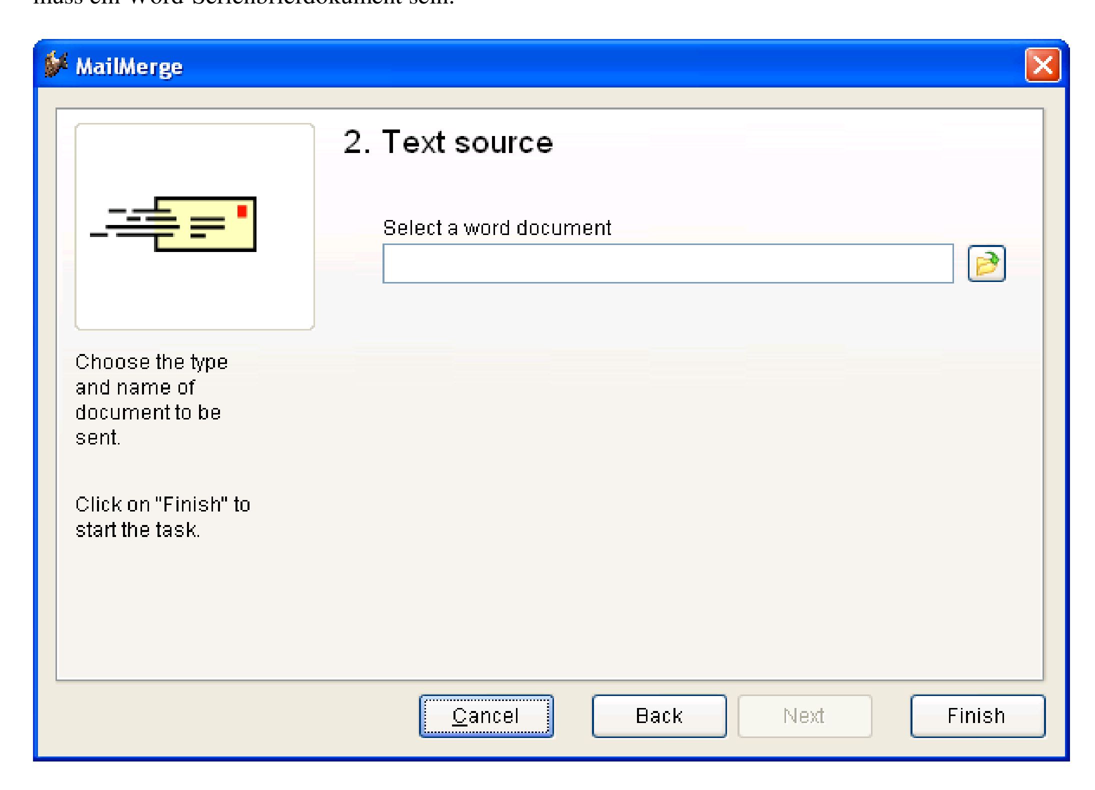
Im nächsten Schritt wählt der Anwender den Quelltext für die zu erstellenden Seriendokumente. Wenn im ersten Schritt E-Mail gewählt wurde, kann der Benutzer im zweiten Schritt zwischen drei Optionen für den E-Mailtext wählen.



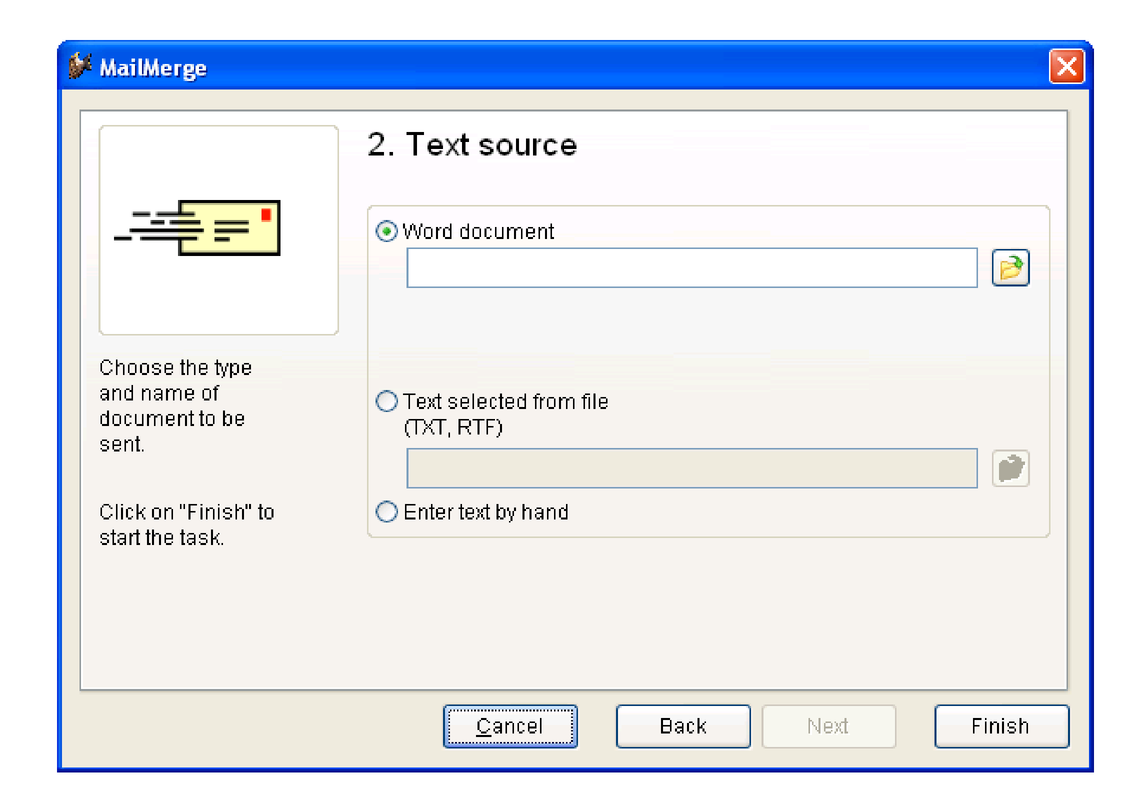
Wenn im ersten Schritt E-Mail oder Fax gewählt wurde, kann ebenfalls zwischen drei Optionen gewählt werden.



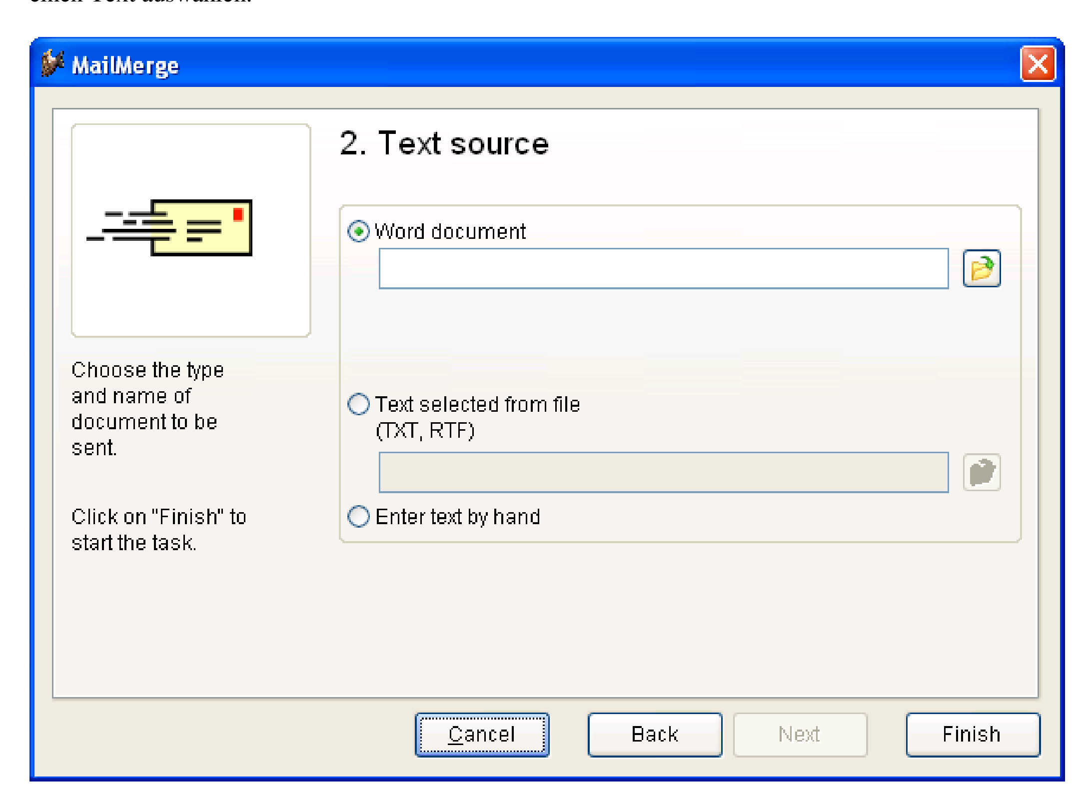
Wenn im ersten Schritt ein Word-Dokument gewählt wurde, kann das Quelldokument auch nur ein Word-Dokument sein. Im zweiten Schritt kann der Benutzer dann das Word-Dokument auswählen. Dieses Dokument muss ein Word-Serienbriefdokument sein.



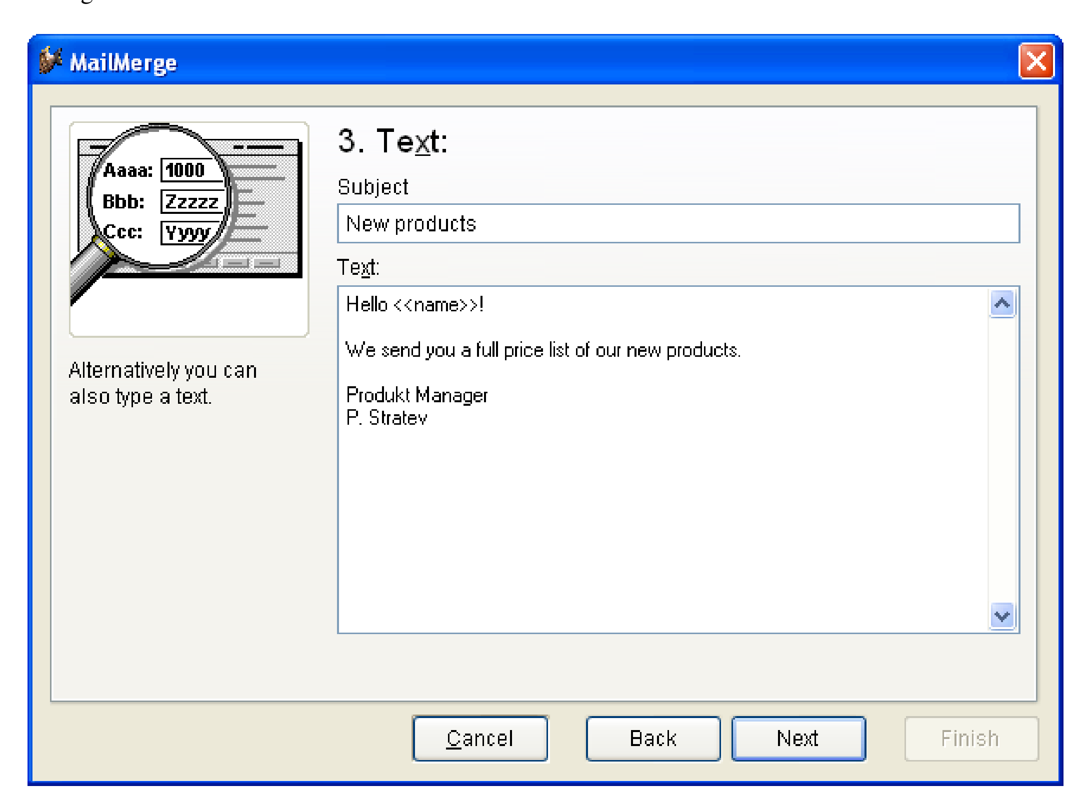
Wenn im ersten Schritt Fax gewählt wurde, kann der Benutzer im zweiten Schritt zwischen drei Optionen für den Fax-Text wählen.



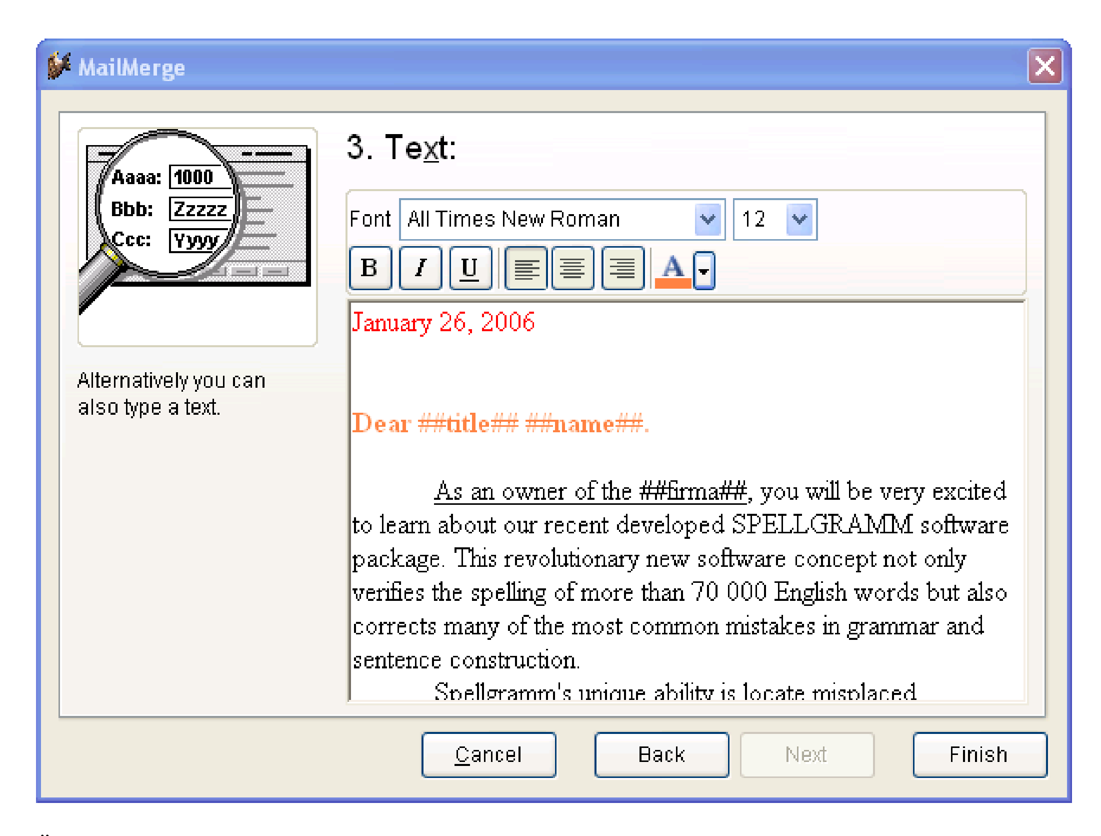
Bei der Auswahl von Drucken im ersten Schritt kann der Benutzer im zweiten Schritt zwischen drei Optionen einen Text auswählen.



Wenn im zweiten Schritt ausgewählt wurde, dass der Text manuell eingegeben werden soll, erscheint im dritten Schritt eine Editbox um einen Text oder ein RTF-Steuerelement um einen RTF-Text eingeben zu können. Zusätzlich kann in diesem Schritt ein Betreff eingegeben werden, wenn im ersten Schritt E-Mail oder Fax oder E-Mail gewählt wurde.

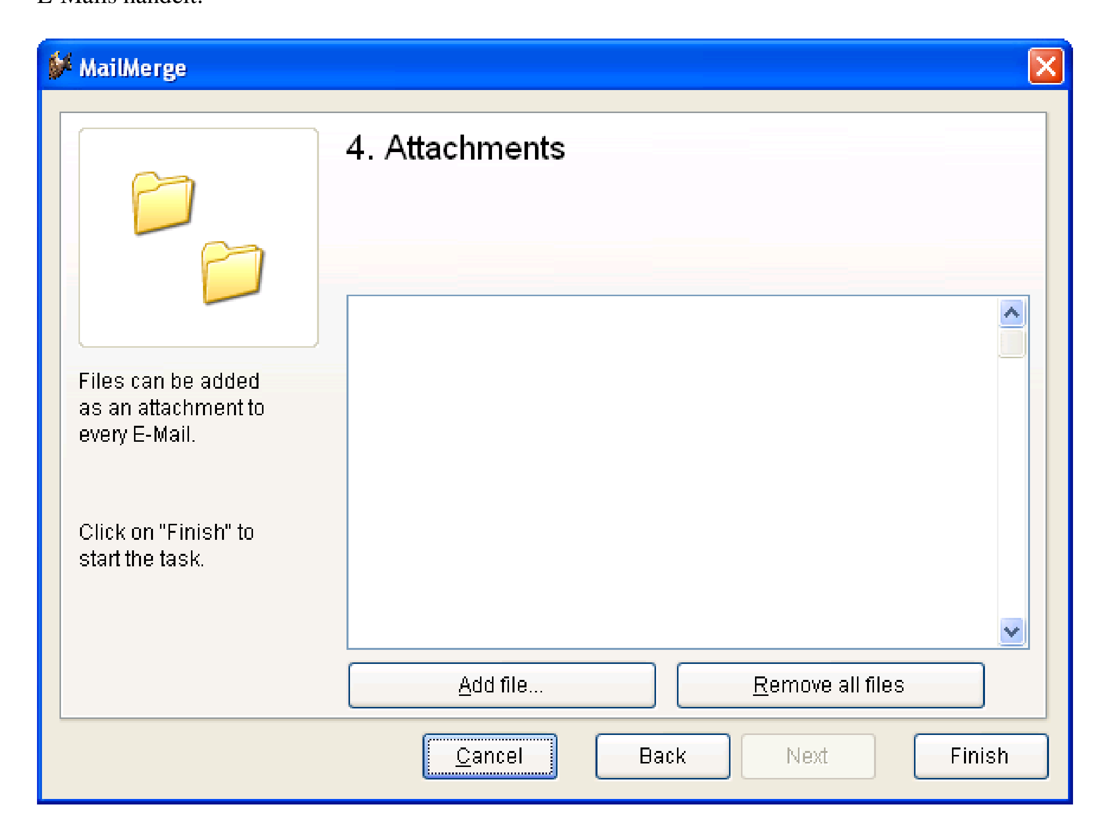


Wenn eine Textdatei als Quelltext ausgewählt wurde, wird der Inhalt hier angezeigt und kann bearbeitet werden. Ausgenommen hiervon sind HTML-Dateien. Änderungen werden im Ausgabedokument berücksichtigt, aber nicht in der Quelldatei oder in der RTF-Datei gespeichert.



Ähnlich wie bei Word-Serienbriefdokumenten können in jedem Quelltext Datenfelder verwendet werden. Die Namen von Datenfeldern müssen in Begrenzungszeichen eingeschlossen sein. Standardmäßig werden die Zeichen „<<“ und „>>“ verwendet. Die Begrenzungszeichen können in den Eigenschaften *cLeftDelim* und *cRightDelim* der Klasse *cMailMerge* bzw. in dem davon abgeleiteten Formular eingestellt werden.

Im nächsten Schritt können Anhänge hinzugefügt werden, wenn es sich bei den zu erstellenden Dokumenten um E-Mails handelt.



Durch einen Mausklick auf die Schaltfläche *Fertig* werden die Seriendokumente erstellt. Im letzten Schritt wird dem Benutzer die Anzahl der erstellten Seriendokumente angezeigt.

23.4.1. Die Klasse cMailMerge

Diese Formularklasse ist in der Klassenbibliothek *Vfxform.vcx* gespeichert.

Mit dieser Klasse können Endanwender anspruchsvolle Seriendokumente erstellen. Folgende Optionen stehen zur Verfügung:

5. E-Mail
   Erstellen von Serien-E-Mails. Der E-Mailtext kann aus einem Word-Dokument oder einer Textdatei stammen oder auch manuell in einer Editbox eingegeben werden. Wenn eine Text-E-Mail erstellt wird, können zusätzlich beliebig viele Dateien als Anhang versendet werden.
6. E-Mail oder Fax
   Wenn eine E-Mailadresse vorhanden ist, wird eine E-Mail versendet. Wenn keine E-Mailadresse vorhanden ist, wird ein Fax versendet.
7. Word-Dokument
   Erstellen einer Word-Serienbriefausgabe basierend auf einem Word-Serienbriefdokument. Die Word-Serienbriefausgabe kann dann in Word beliebig weiterbearbeitet werden.
8. Fax
   Versenden von Serienfaxen basierend auf einem Word-Serienbriefdokument.
9. Drucken
   Drucken von Serienbriefen basierend auf einem Word-Serienbriefdokument oder einem einzugebenden RTF-Text.

Zur Seriendokumenterstellung muss ein Cursor vorhanden sein, der die erforderlichen Felder für jede mögliche Benutzerauswahl enthält.

Eigenschaften

cDataSource
Enthält den Namen der Datenquelle für die Seriendokumentenerstellung. Diese Datenquelle wird von Word oder vom Assistenten direkt verwendet. Alle variablen Felder müssen in dieser Datenquelle enthalten sein.

cMailAddressFieldName
Enthält den Namen des Feldes für die E-Mailadresse. Dieser Feldname muss in dem in *cDataSource* angegebenen Cursor enthalten sein. Diese Eigenschaft wird nur beim Versand von E-Mails verwendet.

cCCFieldName
Enthält den Feldnamen für eine CC E-Mailadresse. Dieser Feldname muss in dem in *cDataSource* angegebenen Cursor enthalten sein. Diese Eigenschaft wird nur beim Versand von E-Mails verwendet.

cBCCFieldName
Enthält den Feldnamen für eine BCC E-Mailadresse. Dieser Feldname muss in dem in *cDataSource* angegebenen Cursor enthalten sein. Diese Eigenschaft wird nur beim Versand von E-Mails verwendet.

cFaxNumberFieldName
Enthält den Namen des Feldes mit der Faxnummer. Dieser Feldname muss in dem in *cDataSource* angegebenen Cursor enthalten sein. Diese Eigenschaft wird nur beim Faxversand verwendet.

cLeftDelim
Linkes Begrenzungszeichen für die Ersetzung von Text. Standardwert ist „<<“. Die hier angegebene Zeichenkette ist die linke Begrenzung eines variablen Wertes.

cRightDelim
Rechtes Begrenzungszeichen für die Ersetzung von Text. Standardwert ist „>>“. Die hier angegebene Zeichenkette ist die rechte Begrenzung eines variablen Wertes.

cMergeText
Nur zur internen Verwendung. Hier ist der Serientext gespeichert.

nEmailsSent
Nur zur internen Verwendung. Zähler für die Anzahl der zu erstellenden Dokumente. Die Anzahl wird auf der letzten Seite des Wizard angezeigt.

nEmailsNotSent
Nur zur internen Verwendung. Zähler für die Anzahl der Dokumente, die nicht versendet werden konnten. Die Anzahl wird auf der letzten Seite des Wizard angezeigt.

nPreviousPageNum
Nur zur internen Verwendung. Enthält die Nummer der vorhergehenden Seite im Wizard.

Methoden

LoadFileContent
Lädt die im zweiten Schritt angegebene Textdatei. Der Text kann im dritten Schritt in einer Editbox bearbeitet werden.

SendMails
Erstellen der Seriendokumente entsprechend der gewählten Optionen. Von hier wird eine der Methoden *SendThroughMapi* oder *SendThroughOleWord* aufgerufen.

SendThroughMapi
Erstellt Serien-E-Mails unter Verwendung der VFX-Klasse *cEmail*.

SendThroughOleWord
Diese Methode erstellt Serienbriefe per OLE-Automatisierung von Word. Auf diesem Weg kann die Serienbriefausgabe in ein Word-Dokument gespeichert, per Fax gesendet, gedruckt oder per E-Mail versendet werden.

23.5. Erweitertes Bearbeitungsprotokoll

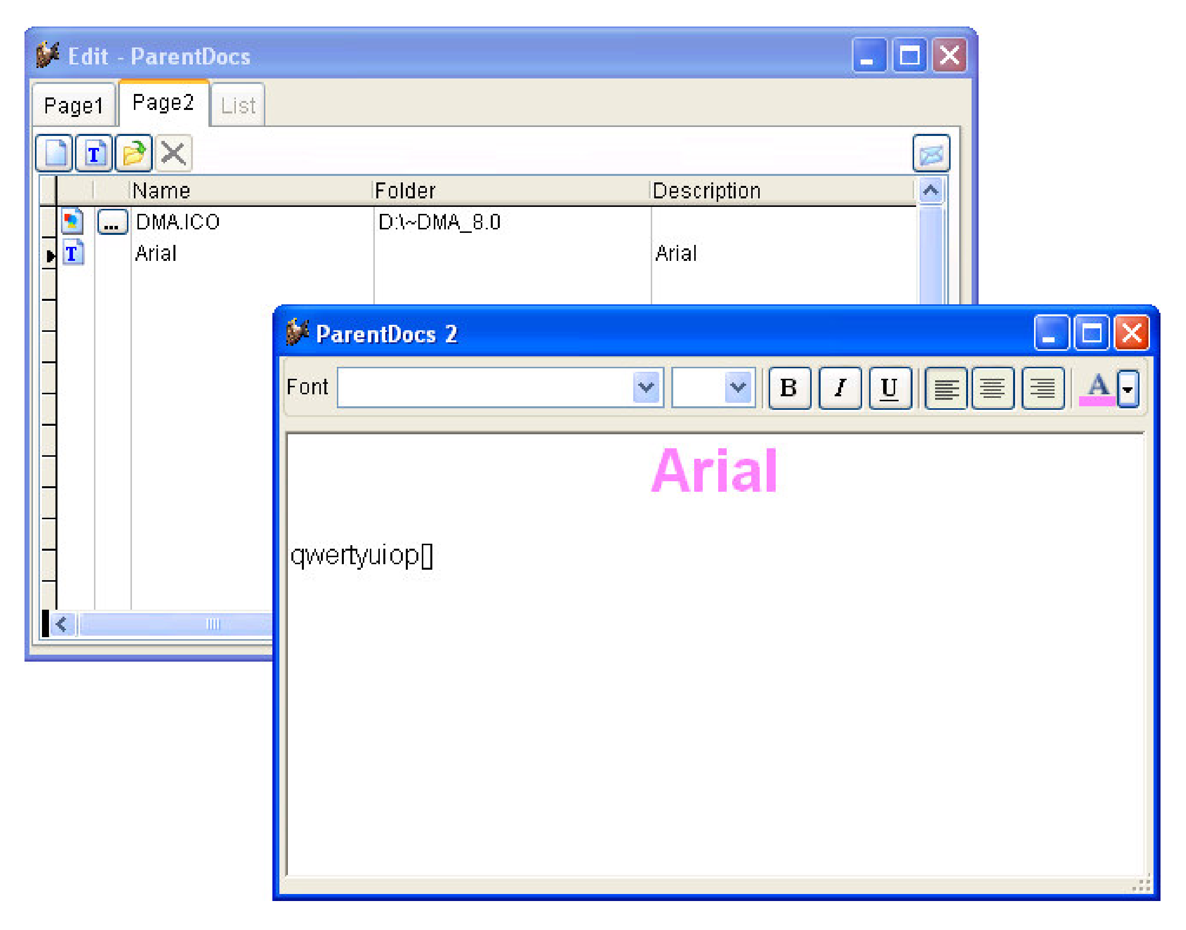
Im Menü von erstellten Anwendungen gibt es das neue Untermenü *Extras, Bearbeitungsprotokoll*.

In diesem Untermenü gibt es die Menüpunkte *Bearbeitungsprotokoll* und *Bearbeitungsinformationen*. Im Formular *Bearbeitungsprotokoll* werden Informationen zum aktuellen Datensatz angezeigt. Im Formular *Bearbeitungsinformationen* werden Informationen für alle Datensätze der Tabelle angezeigt. Dieses Formular bietet alle Möglichkeiten eines normalen VFX-Datenbearbeitungsformulars, einschließlich Suche sowie Druck- und Exportmöglichkeit. Um Bearbeitungsinformationen anzeigen zu können, muss ein Formular geöffnet sein.

23.6. Dokumentenverwaltung

Die Klasse *CDocumentManagement* dient zur Verwaltung von Dokumenten aller Art (z. B. Word, Excel, Powerpoint) innerhalb einer Anwendung. Die Klasse *CDocumentManagement* ist ein Container, der Child-Datensätze zum aktuellen Datensatz im Formular verwaltet. Die Dokumentenverwaltung ermöglicht dem Anwender Dokumente zu öffnen und als E-Mailanhang zu versenden. Es ist auch möglich RTF-Texte zu verwalten und zu bearbeiten.

Diese Klasse kann bestehenden Formularen einfach hinzugefügt werden.



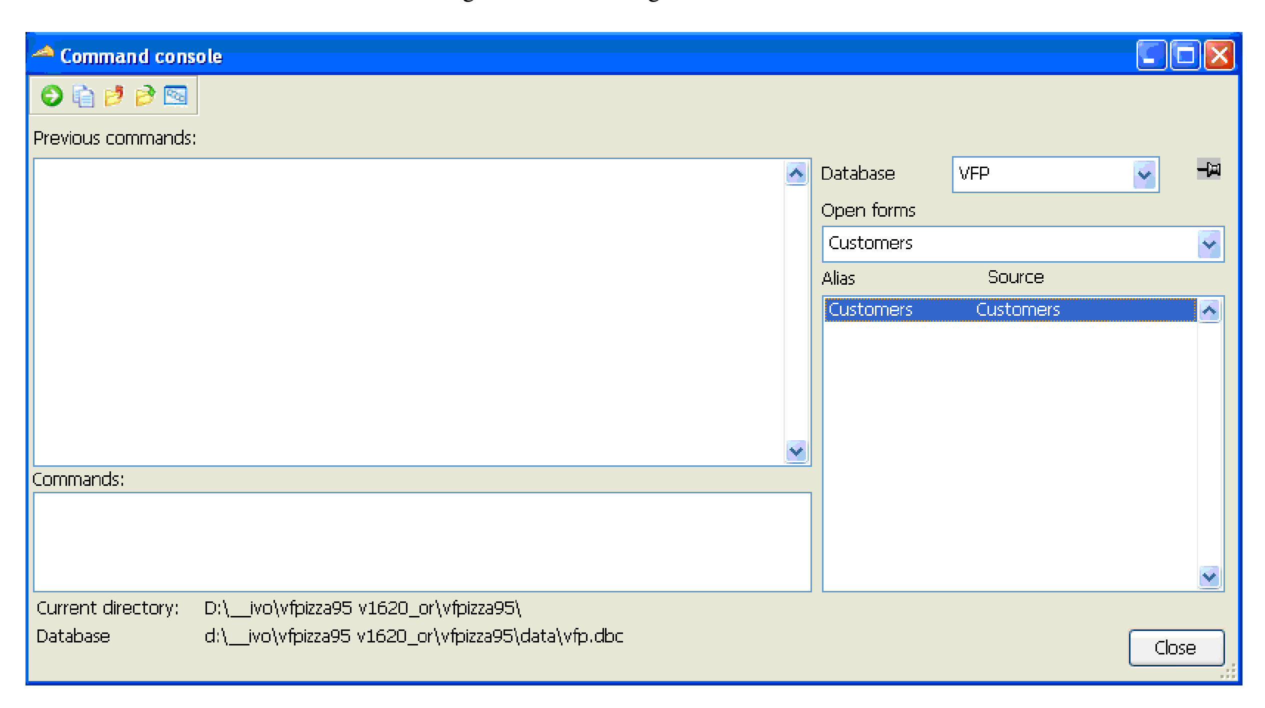
Das RTF-Bearbeitungsformular wird als Child-Formular geöffnet, wenn in der Dokumentverwaltung ein RTF-Text ausgewählt ist. Mit der Schaltfläche Neu RTF wird ein neues, leeres RTF-Dokument angelegt. Wenn in der Dokumentenverwaltung ein RTF-Dokument ausgewählt ist und der Benutzer auf die Schaltfläche *Öffnen* klickt, wird das RTF-Bearbeitungsformular geöffnet. Wenn der Benutzer auf die Schaltfläche *E-Mail* klickt, wird das RTF-Dokument in einer Datei gespeichert und die Datei wird als E-Mailanhang versendet.

Wenn ein RTF-Eintrag in der Dokumentverwaltung gelöscht wird, wird der RTF-Text in der Tabelle *VFXRTF* ebenfalls gelöscht.

Wenn der Parent-Datensatz gelöscht wird, werden alle Einträge in der Dokumentenverwaltung und alle dazugehörenden RTF-Texte gelöscht.

23.7. VFX Befehlseingabe

Für Benutzer mit Administratorrechten steht der Dialog zur Befehlseingabe zur Verfügung. Hiermit kann der Administrator zur Laufzeit einer Anwendung VFP Befehle eingeben.



Im VFX – Application Builder kann mit der Option *Enable Command Console* eingestellt werden, ob der Dialog zur Befehlseingabe Benutzern mit Administratorrechten zur Verfügung stehen soll. Zur Laufzeit kann der Dialog aus dem Menü *Extras* gestartet werden.

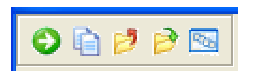
Die auszuführenden Befehle werden in der Editbox *Befehl* eingegeben. Ein eingegebener Befehl kann mit der Eingabetaste oder mit einem Klick auf die Schaltfläche *Ausführen* am oberen Dialogrand ausgeführt werden. Eine Historie aller ausgeführten Befehle wird in der Listbox *Bisherige Befehle* angezeigt. Bereits ausgeführte Befehle können mit einem Rechtsklick aus der Listbox in die Textbox zur Befehlseingabe kopiert werden.

Die Combobox *Datenbank* ermöglicht die Auswahl einer Datenbank aus allen zurzeit geöffneten Datenbanken. In der Combobox *Geöffnete Formulare* wird eine Liste aller zurzeit geöffneten Formulare angezeigt. Durch die Auswahl eines Formulars wird die Datensitzung auf die Datensitzung des Formulars gewechselt und in der Listbox *Alias Source* werden die in dieser Datensitzung geöffneten Cursor angezeigt.

Am unteren Rand des Dialogs werden der aktuelle Ordner und die aktuell geöffnete Datenbank angezeigt.

Mit dem grafischen Kontrollkästchen *Always on top* wird die Eigenschaft *AlwaysOnTop* des Dialogs eingestellt. Wenn dieses Kontrollkästchen markiert ist, erscheint dieser Dialog als oberstes Formular.

Am oberen Rand des Dialogs befindet sich eine Gruppe von Schaltflächen.

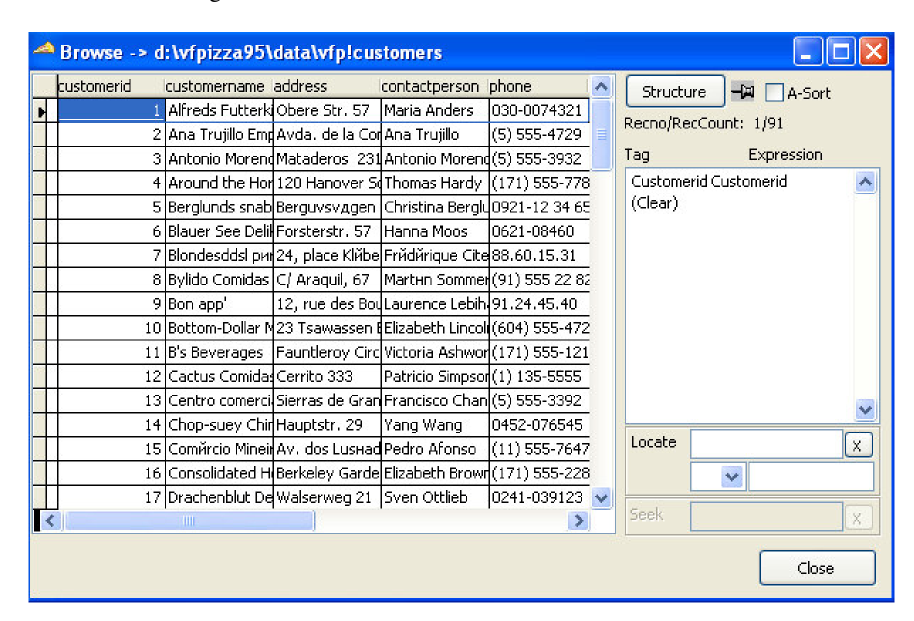


Ausführen	Ausführen des Befehls in der Editbox *Befehl* oder Ausführen des selektierten Eintrags in der Listbox *Bisherige Befehle*.
Kopieren	Kopiert den selektierten Befehl aus der Listbox *Bisherige Befehle* in die Editbox *Befehl*.
Use	Schließen des aktuellen Cursors (entspricht der Ausführung des Befehls *USE*).

Use ? in 0 Öffnet eine Tabelle in einem neuen Arbeitsbereich (entspricht der Ausführung des Befehls *USE ? IN 0*).

Browse Öffnet das VFX Browse Formular für den aktuell selektierten Arbeitsbereich.

Wenn der Befehl *BROWSE* ausgeführt wird oder wenn die Schaltfläche *Browse* gedrückt wird, wird das VFX Browse Formular geöffnet.



Die Daten des aktuell ausgewählten Arbeitsbereichs werden im Grid auf der linken Seite des Formulars angezeigt. Auf der rechten Seite des Formulars befinden sich zusätzliche Steuerelemente:

Structure Diese Schaltfläche zeigt die Struktur des Cursors im aktuellen Arbeitsbereich an (entspricht der Ausführung des Befehls *MODIFY STRUCTURE*).

Always on top Stellt die Eigenschaft *AlwaysOnTop* des Formulars ein.

A-Sort Wenn dieses Kontrollkästchen markiert ist, werden die Spalten im Grid in alphabetischer Reihenfolge angeordnet.

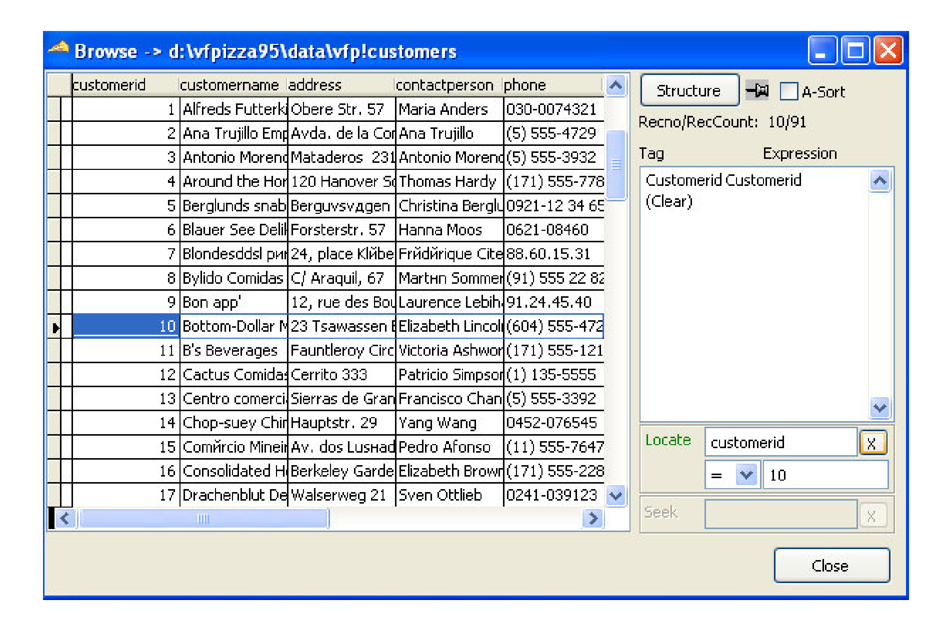
RecNo/RecCount Anzeige der Satznummer des aktuellen Datensatzes sowie der Anzahl der Datensätze.

Tags list Hier werden alle Indexschlüssel zum aktuellen Arbeitsbereich angezeigt. Durch einen Doppelklick auf einen Eintrag werden die Daten im Grid entsprechend diesem Index sortiert angezeigt.

Locate / Seek Hiermit kann nach einem Ausdruck gesucht werden.

Locate / Seek

Wenn mit den Befehlen *LOCATE* oder *SEEK* ein Datensatz gefunden wird, wird die Bezeichnung in grüner Schrift angezeigt. Wenn die Suche nicht erfolgreich ist, wird die Bezeichnung mit roter Schrift angezeigt.



Um den Befehl *SEEK* verwenden zu können, muss zuvor ein Indexschlüssel ausgewählt werden. Wenn kein Indexschlüssel aktiv ist, ist die Option *SEEK* deaktiviert. Der aktuelle Indexschlüssel kann mit einem Doppelklick in die Listbox *Tag Expression* gewechselt werden. Der aktuelle Indexschlüssel wird unterhalb des Grids angezeigt.

23.8. Die Klasse cGridMover

Die Klasse *cGridMover* funktioniert so ähnlich wie die Klasse *cMover*. Der Unterschied besteht darin, dass die Klasse *cGridMover* mit zwei Grids statt zwei Listboxen arbeitet. Dadurch stehen in diesem Mover-Dialog alle Funktionen zur Verfügung, die VFX Grids standardmäßig bieten, wie Sortierung und inkrementelle Suche.

Das Grid auf der linken Seite im Dialog enthält alle zur Auswahl stehenden Daten. Das Grid auf der rechten Seite enthält die Liste der ausgewählten Elemente. Der Benutzer kann jede beliebige Anzahl von Elementen mit den Pfeiltasten auswählen oder auch aus der Auswahl entfernen.

Die Felder in den Arbeitsbereichen für die auswählbaren und ausgewählten Elemente müssen die gleichen Feldnamen haben. Zusätzlich ist ein Feld erforderlich, das intern verwendet wird, und anzeigt, welche Datensätze ausgewählt sind. Dieses Feld sollte nicht im Grid angezeigt werden. Der Name dieses Feldes wird in der Eigenschaft *cControlFieldName* gespeichert. Dieses Feld muss vom Typ numerisch oder logisch sein und wird von der Klasse *cGridMover* zur Steuerung verwendet.

Eigenschaften

cSourceAlias — Name des Alias für das Auswahlgrid auf der linken Seite.

nControlFieldNameType — Nur zur internen Verwendung. Hier wird der Feldtyp des Feldes *cControlFieldName* gespeichert. 1 - Logisch, 2 - Numerisch.

Wenn diese Klasse auf einem Formular verwendet wird, müssen auch die Recordsource sowie die Controlsources der Spalten der beiden Grids eingestellt werden.

Methoden

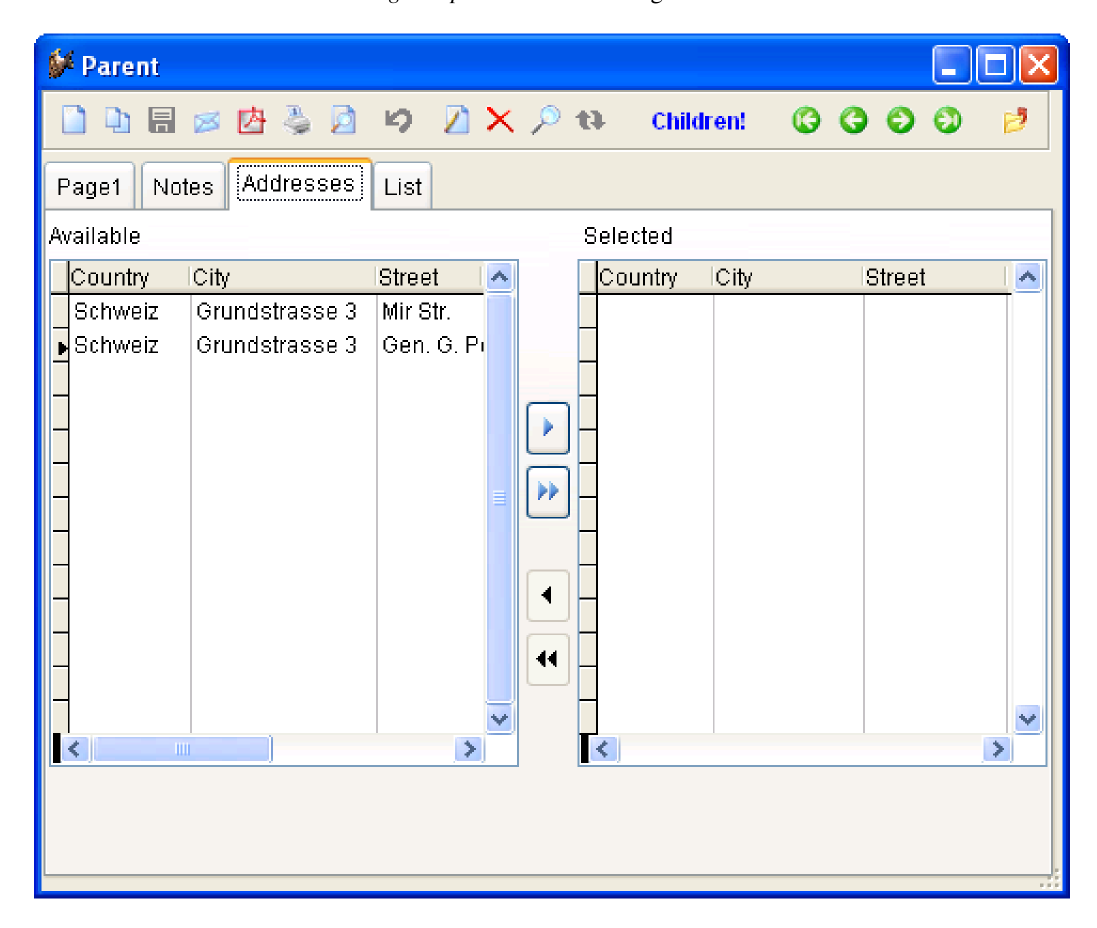
onPostInsert — Diese Methode wird ausgeführt, wenn ein Datensatz ausgewählt wird.

onPostSave — Diese Methode wird nach dem Speichern der Daten des Formulars ausgeführt.

onPreSave — Diese Methode wird vor dem Speichern der Daten des Formulars ausgeführt.

onUndo — Diese Methode wird von der Methode *onUndo* des Formulars aufgerufen.

RefreshMoverButtons	Durch Aufruf dieser Methode wird der Status der Schaltflächen aktualisiert.
RefreshSourceList	Aktualisieren der Auswahlliste entsprechend der Liste mit den ausgewählten Werten. Diese Methode wird von der Methode *onRecordMove* des Formulars aufgerufen.
LangSetUp	Einstellen der sprachspezifischen Texte. Diese Methode wird von der Methode *LangSeTup* des Formulars aufgerufen.



Wenn Datensätze ausgewählt werden, werden die Daten des Datensatzes (Datenquelle ist die Recordsource des Grid mit den auswählbaren Daten) in einen neuen Datensatz in den Arbeitsbereich mit den ausgewählten Datensätzen (Datenquelle ist die Recordsource des Grid mit den ausgewählten Daten) geschrieben. Es werden die Inhalte aller Felder mit identischen Namen kopiert, auch wenn diese nicht in den Grids angezeigt werden. Wenn ein Datensatz ausgewählt wird, wird dieser nicht mehr im Auswahlgrid angezeigt.

Wenn Datensätze aus der Auswahl entfernt werden, werden diese aus dem Arbeitsbereich mit den ausgewählten Daten gelöscht und wieder in der Auswahlliste angezeigt.

Der folgende Beispiel-Code kann verwendet werden, um mit einem Doppelklick einen Datensatz auszuwählen.

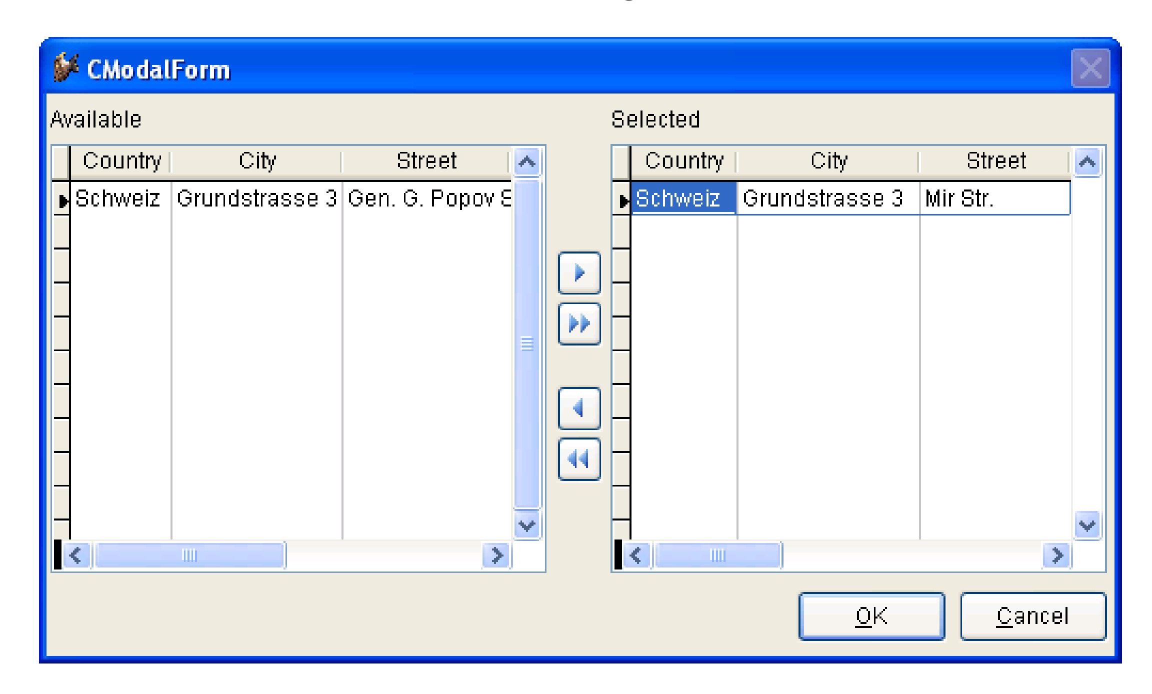
Für die Methode *DblClick* von Textboxen im Auswahlgrid:

```
This.Parent.Parent.Parent.cmdAdd.Click()
```

Für die Methode *DblClick* von Textboxen im Grid mit den ausgewählten Datensätzen:

```
This.Parent.Parent.Parent.cmdRemove.Click()
```

23.9. Die Klasse cGridMoverDialog



Die Klasse *cGridMoverDialog* ist ein Dialog basierend auf der Klasse *cModalForm,* der ein *cGridMover* Steuerelement enthält. Dieser Dialog bietet die Funktionalität der Klasse *cGridMover* in einem Dialog.

Wenn die Klasse *cGridMoverDialog* verwendet wird, wird erwartet, dass die Daten im aufrufenden Formular in einem Grid angezeigt werden. Eine Referenz auf dieses Grid wird dem Grid-Mover-Dialog als Parameter übergeben. Das Grid wird nach Beenden des Grid-Mover-Dialogs automatisch aktualisiert.

Parameter

tcSourceAlias	Aliasname des Cursors mit den auswählbaren Daten. Dieser Aliasname wird die Recordsource des Grid auf der linken Seite.
tcDestinationAlias	Aliasname des Cursors mit den ausgewählten Daten.
tcControlField	Name des Feldes, das verwendet wird, um die Auswahl zu kennzeichnen.
toGridDestination	Referenz auf das Grid im aufrufenden Formular. Dieser Parameter kann auch leer bleiben.
tcCommaSeparatedFieldList	Komma-separierte Liste von Feldnamen. Diese Feldnamen werden für die Controlsources der Spalten in beiden Grids verwendet.
tcCommaSeparatedHeaderList	Komma-separierte Liste von Spaltenüberschriften. Diese Spaltenüberschriften werden für beide Grids verwendet.
tcCommaSeparatedColumnWidth	Komma-separierte Liste mit numerischen Werten zur Einstellung der Spaltenbreiten in beiden Grids.

Eigenschaften

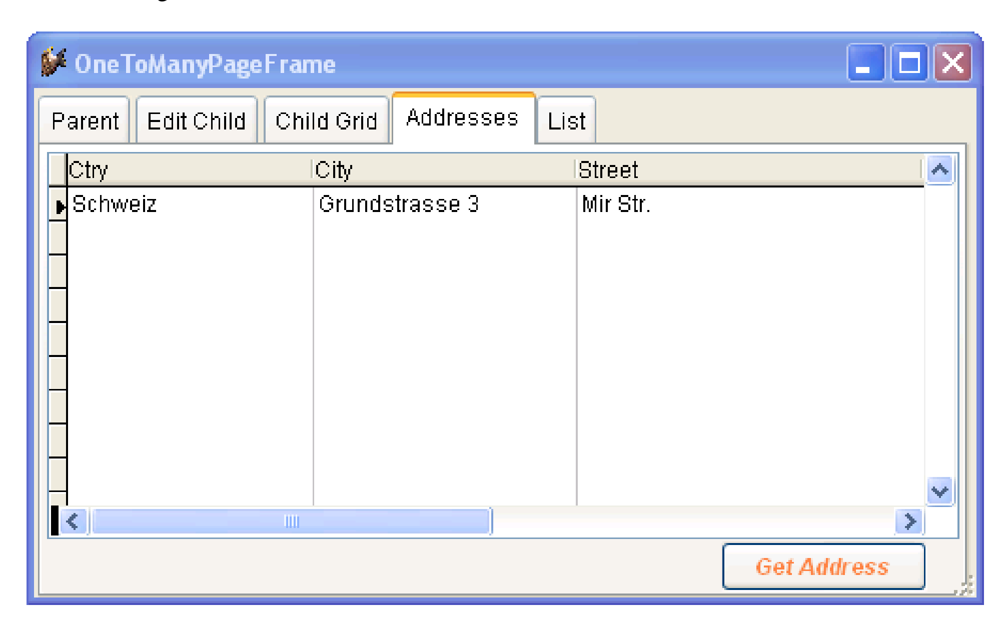
oGridDestination	Enthält eine Referenz auf das Grid im aufrufenden Formular, das nach der Auswahl aktualisiert werden muss.

Methoden

SetDestinationData Aktualisiert den Arbeitsbereich mit den ausgewählten Daten.

Hier ein Beispiel, wie die Klasse *cGridMoverDialog* in der Praxis verwendet werden kann.

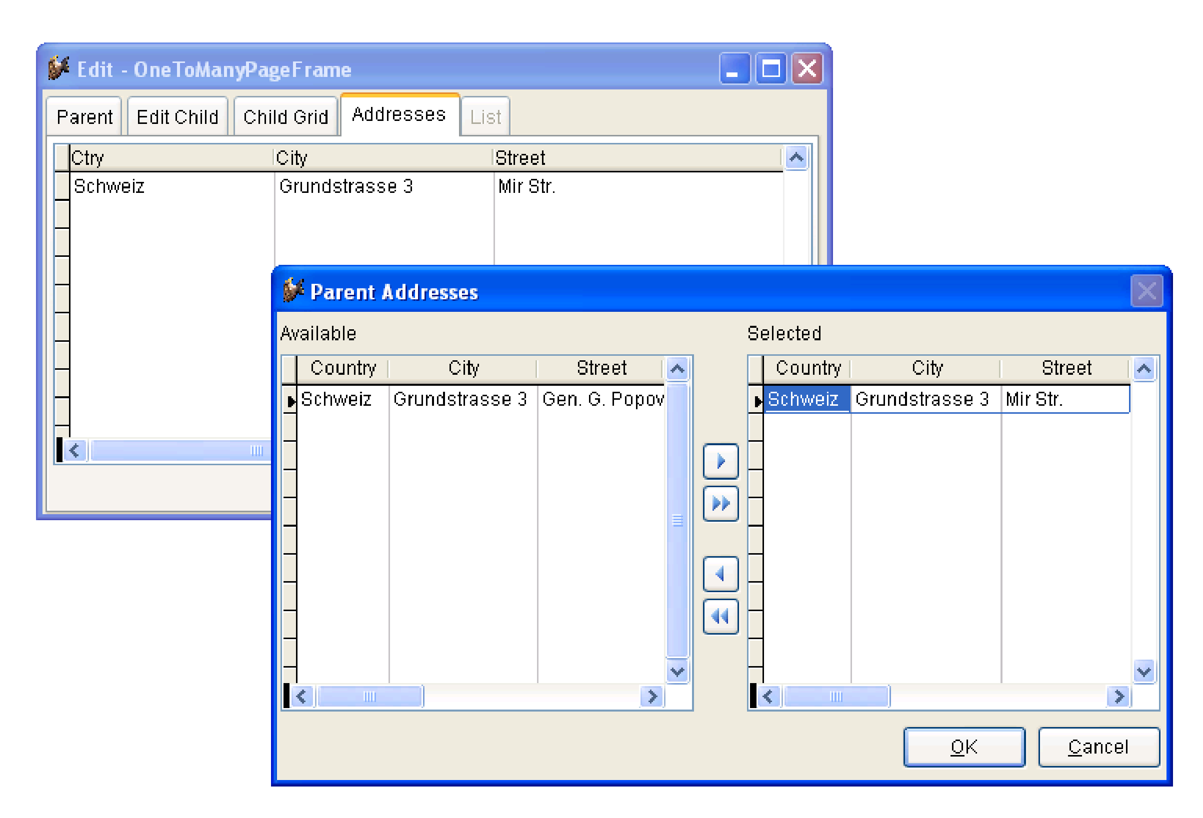
Nehmen wir an, wir haben ein Onetomany-Formular und die Child-Daten sollen in einem Mover-Dialog ausgewählt werden. Auf dem aufrufenden Onetomany-Formular wird eine Schaltfläche platziert, die den Grid-Mover-Dialog aufruft.



Hier der Code aus dem *Click* Ereignis der Schaltfläche *Get Address*:

```
Local loGridMover
loGridMover =  CREATEOBJECT("cGridMoverDialog","caAddress",
"caParentAddress", "Selected", ThisForm.pgfPageFrame.Page4.Cchildgrid1,
"ctry,city,street", "Country,City,Street", "100,120,140")
loGridMover.Caption = "Parent Addresses"
loGridMover.Show()
```

Nach dem Instanziieren des Objekts *loGridMover* besteht die volle Kontrolle über dieses Steuerelement und es können alle Eigenschaften nach Wunsch eingestellt werden und es können alle Methoden ausgeführt werden. Das *Show* Ereignis zeigt den modalen Dialog an und die Code-Ausführung im *Click* Ereignis wird erst dann fortgesetzt, wenn der Benutzer den Grid-Mover-Dialog schließt.



Wenn der Dialog gestartet wird, werden im Grid mit den ausgewählten Daten die gleichen Datensätze angezeigt, die auch im aufrufenden Formular im Childgrid zu sehen sind. Das Grid mit den auswählbaren Daten enthält alle Datensätze des Arbeitsbereiches mit den auswählbaren Daten, ausgenommen sind die bereits ausgewählten Datensätze.

Im Grid-Mover-Dialog kann der Benutzer Datensätze nach Belieben auswählen. Wenn der Benutzer auf die Schaltfläche *OK* klickt, werden die ausgewählten Datensätze in den Arbeitsbereich des aufrufenden Formulars geschrieben. Die Änderungen im Grid-Mover-Dialog werden verworfen, wenn der Benutzer auf die Schaltfläche *Abbrechen* klickt. Wenn die Datensätze in den Arbeitsbereich des aufrufenden Formulars geschrieben werden, wird für jeden Datensatz die Methode *onPostInsert* des Grids im aufrufenden Formular ausgeführt.

24. Kleine Erweiterungen

1. Wenn in Onetomany-Formularen mit CursorAdaptern gearbeitet wird, wird der Code für die Methode *OnPostInsert()* von Childgrids nur dann generiert, wenn die Werte der Eigenschaften *ForeignKeyName* und *ForeignKeyValue* des Child-Arbeitsbereiches nicht leer sind.
2. Bei der Aktualisierung der Struktur von Kundendatenbanken werden Tabellen automatisch nicht berücksichtigt, die sich sowohl im Datenordner befinden, als auch in die Exe-Datei eingeschlossen sind.
3. Bei der Aktualisierung der Struktur von SQL Server Datenbanken werden automatisch in allen hinzuzufügenden Feldern NULL Werte erlaubt, wenn kein Standardwert zugewiesen werden soll.
4. Die Aktualisierung der Struktur von Datenbanken wird nur dann ausgeführt, wenn die Anwendung als Exe-Datei gestartet wird (VERSION(2) <> 2).
5. Die Breiten der Spalten in Comboboxen von der Klasse *cComboPickList* werden nur dann automatisch berechnet, wenn der Wert der Eigenschaft *lAutoAdjustColumnWidths* auf *.T.* eingestellt wird.
6. Die Textbox für den Schriftschnitt im Grid-Berichtsdialog ist jetzt lokalisiert.
7. Die Eigenschaften *goProgram.cCompanyName* und *goProgram.cAppName* werden verwendet um einen Ordner unter *Dokumente und Einstellungen\AllUsers\Firmenname\Anwendungsname* anzulegen, wenn der aktuelle Benutzer das Recht hat, diesen Ordner anzulegen. Wenn der aktuelle Benutzer dieses Recht nicht besitzt, wird ein Ordner unterhalb von *Eigene Dateien\Firmenname\Anwendungsname* angelegt. In diesem Ordner werden die Tabellen *Vfxacomp.dbf* und *Vfxpath.dbf* sowie die Datei *Vfx.ini* gespeichert. Wenn die Werte dieser Eigenschaften leer sind, werden diese Dateien im Ordner der Exe-Datei gespeichert.
8. Der VFX – Application Builder führt für die Werte aller Eigenschaften vor dem Speichern ALLTRIM() aus.

25. DB/2 Unterstützung
Specifics when working with DB2 UDB

- GetSQLServers() and GetSQLDatabases() functions in VFX.fll, has been enlarged to receive driver name as a parameter when retrieving available servers and databases.

- An identifier that does not comply with the rules for the format of regular identifiers (for instance table and field names containing spaces) must always be delimited

  In SQL Server, delimited identifiers can be quoted (“”) or bracketed ([]). To work independent on QUOTED_IDENTIFIER setting, VFX always uses bracketed identifiers when working with SQL Server database.
  DB2 UDB uses only quoted identifiers.
  In VFX the database type is determined, based on used driver type and appropriate delimiters are used.

- Generated autoincrement value in DB2 UDB can be retrieved using the function *IDENTITY_VAL_LOCAL()*. The InsertCmdRefreshCmd is generated in *cBaseDataAccess.GetInsertRefreshCmd()* depending on database source type

- Strings cannot be concatenated with + operation (see Strings processing below)

- When connected to the database as a specific user, only objects that belong to this user or to his schema are accessible. However SQLTABLES() function, which is used by builders, returns a list of all tables in the database regardless of the schema they belong to.

26. VFP, SQL Server and DB2 UDB data types

VFP data type	SQL Server data type	DB2 UDB data type	Range of values
CHAR(n)	CHAR(m)	CHAR(n)	1 <= *m*<= 8000 1 <= *n* <= 254
VARCHAR(k)	VARCHAR(m)	VARCHAR(n)	1 <= *m*<= 8000 1 <= *n* <= 32762 1 <= k <=254
	LONG	VARCHAR(n)	if *n* <= 32700 bytes
MEMO	TEXT	CLOB(2GB)	if *n* <= 2 GB
	TINYINT	SMALLINT	- 32768 to 32767
	SMALLINT	SMALLINT	- 32768 to 32767
INT	INT INTEGER	INT INTEGER	$- 2^{31}$ to $(2^{31} - 1)$
	BIGINT	BIGINT	
DECIMAL(p,s)	DEC(p,s) DECIMAL(p,s)	DEC(p,s) DECIMAL(p,s)	$- (10^{31}+1)$ to $(10^{31} - 1)$ (p+s <= 31)
NUMERIC(q,r)	NUMERIC(p,s)	NUM(p,s) NUMERIC(p,s)	(p+s <= 31) $- (10^{31}+1)$ to $(10^{31} - 1)$ (q+r <= 20) $- (10^{19}+1)$ to $(10^{20} - 1)$
FLOAT(q,r)	FLOAT(p)	FLOAT(p)	
	REAL	REAL	
DOUBLE	DOUBLE	DOUBLE PRECISION	
LOGICAL	BIT	CHAR(1) FOR BIT DATA	0 or 1
CHAR BINARY(n)	BINARY(m)	CHAR(n) FOR BIT DATA	1 <= *m*<= 8000 1 <= *n* <= 254
VARBINARY(k)	VARBINARY(m)	VARCHAR(n) FOR BIT DATA	1 <= *m*<= 8000 1 <= *n* <= 32672 1 <= k <=255
GENERAL	IMAGE	BLOB(n)	if *n* <= 2 GB
	NTEXT	DBCLOB(n)	0 <= *n* <= 2 GB
	SMALLDATETIME	TIMESTAPMP	Jan 1, 0001 to Dec 31, 9999
DATETIME	DATETIME	TIMESTAPMP	Jan 1, 0001 to Dec 31, 9999
	TIMESTAMP	CHAR(8) FOR BIT DATA	
DATE		DATE (MM/DD/YYYY)	year: 0001 to 9999 month: 1 to 12 day: 1 to 31
		TIME (HH24:MI:SS)	hour: 0 to 24 minutes: 0 to 60 seconds: 0 to 60
	NCHAR(m)	GRAPHIC(n)	1 <= *m*<= 4000 1 <= *n* <= 127
	NVARCHAR(m)	VARGRAPHIC(n)	1 <= *m*<= 4000 1 <= *n* <= 16336
	LONG	VARGRAPHIC(n)	1 <= *n* <= 16336
	SMALLMONEY	NUMERIC(10,4)	
CURRENCY	MONEY	NUMERIC(19,4)	
CHAR(32)	UNIQUEIDENTIFIER	CHAR(13) FOR BIT DATA	

27. Builders and VFX features considerations

27.1. CursorAdapter wizard

SQLTABLES() function used to obtain list of database tables. Returns a cursor with all available tables in the database. However, later user can access only tables, which belong to his schema

27.2. Upsizing wizard

Upsizing wizard should be enlarged to convert VFP database objects to DB/2
A possible difficulties may arise when in VFP database are created two indexes on the same field in a table using different VFP expressions, like:

```
LOWER(LEFT(kdstrasse,7))
UPPER(kdstrasse)
(goma.dbc, mag.dbf, Gothaer project)
```

When these two indexes are upsized to MS SQL Server the functions are removed, as far as SQL Server does not support index expressions. In that case in SQL Server database are created two identical indexes with different names. This is not possible in DB2 UDB. In such case in DB2 database only the first index will be created.
FillDatadictForConnection() function uses *SQLTABLES()* to creates a cursor with info about database and tables. However when executin the function toward DB2 database, table names are retrieved, but the column Table_Cat contains only *NULL* values Instead of Database name. A workaround can be retrieving database name from the connection string.
ReadSQLTables() function uses bracketed identifiers. It must be changed to appropriate delimiter, depending on database engine type.

27.3. Client database update

Now the table Datadict holds structure information in SQL Server specific format. In order to support DB/2, it will be necessary in Metadata wizard to add functionality to gather structure information in DB/2 specific format and in application to add functionality to apply DB/2 structure information toward DB/2 database.

28. Application programming considerations

28.1. Data types conversion considerations

The automatic type's conversion, used by Cursor adapter classes helps to overcome differences between almost all data types in VFP and DB/2. The only problems appear with following data types, as far as their correspondent types are incompatible:

VFP/MS SQL	**DB2 UDB**
LOGICAL/BIT	CHAR(1) bit data
GENERAL/IMAGE	BLOB

When trying to convert a column with data type CHAR(1) bit data to VFP data type *L*, using the conversion feature in CursorAdapter class by setting *Use cursor schema* = *.T.* and placing *L* as data type in the CursorSchema, on CursorFill execution is raised the following error : Type conversion required by the DataType property for field <*field name*> is invalid.

For more detailed information, see: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dv_foxhelp9/html/c101845f-d0a1-4f86-b1ba-225929032da6.asp

28.2. SQL language syntax and semantics

28.2.1. Column and table aliases

In same way as in VFP and SQL Server, DB2 UDB syntax uses the *AS* keyword to define column aliases and table correlation names, as shown below:

```
SELECT d.deptno AS DepartmentNo,
e.empno AS EmployeeNo,
e.firstnme || ' ' || lastname AS EmployeeName
   FROM department AS d INNER JOIN employee AS e
ON d.deptno = e.workdept
```

Column names and table names, containing spaces, are enclosed in “”

28.2.2. Functions

Some often used functions have same names in VFP, SQL Server and DB/2 UDB

UPPER()
LEFT()
RIGHT()
LTRIM()
RTRIM()

28.2.3. Strings processing

In difference of VFP and SQL Server, string cannot be concatenated using + operation. For this purpose is used || or *CONCAT*

```
SELECT d.deptno AS DepartmentNo,
e.empno AS EmployeeNo,
e.firstnme || ' ' || lastname AS EmployeeName
   FROM department AS d INNER JOIN employee AS e
ON d.deptno = e.workdept
```

28.2.4. Datetime processing functions

VFP	SQL Server	DB2 UDB
YEAR(DATETIME())	DATEPART(year, GETDATE())	YEAR(CURRENT DATE)
QUARTER(DATETIME())	DATEPART(quarter, GETDATE())	QUARTER(CURRENT DATE)
MONTH(DATETIME())	DATEPART(month, GETDATE())	MONTH(CURRENT DATE)
	DATEPART(dayofyear, GETDATE())	DAY OFYEAR(CURRENT DATE)
DAY(DATETIME())	DATEPART(day, GETDATE())	DAY(CURRENT DATE)
WEEK(DATETIME())	DATEPART(week,	WEEK(CURRENT TIME)

	GETDATE())	
DOW(DATETIME())	DATEPART(weekday, GETDATE())	DAYOFWEEK(CURRENT DATE)
HOUR(DATETIME())	DATEPART(hour, GETDATE())	HOUR(CURRENT TIME)
MINUTE(DATETIME())	DATEPART(minute, GETDATE())	MINUTE(CURRENT TIME)
	DATEPART(second, GETDATE())	SECOND(CURRENT TIME)
	DATEPART(millisecond, GETDATE())	MICROSECOND(CURRENT TIMESTAMP)

28.2.5. NULL values

In where clause for VFP, SQL Server and DB2 UDB the syntax *<field name> IS NULL* or *<field name> IS NOT NULL* is used in same way

28.2.6. Unqualified columns

SQL Server and VFP permit the use of an unqualified column wildcard (*) alongside other elements in the *SELECT* clause list. DB2 UDB adheres to the SQL standard which states that a *SELECT* element that contains an unqualified column wildcard cannot contain anything else. DB2 UDB requires these elements to be qualified (a sequence of t1.*, t2.*, ... where t1, t2, ... are the tables in the *FROM* clause).
For example, the following query is valid in VFP and SQL Server, but not in DB2 UDB:

```
SELECT e.*, * FROM employee e, jobs j WHERE e.job_id = j.job_id
```

Because an unqualified column wildcard appears in the SELECT list alongside other elements, the query would be invalid in DB2 UDB. To convert this query, the wildcard column would need to be qualified, like the following modified query demonstrates:

```
SELECT e.*, j.* FROM employee e, jobs j WHERE e.job_id = j.job_id
```

28.2.7. SELECT INTO

The VFP's *SELECT INTO TABLE* and SQL Server's *SELECT INTO* statement is completely different than DB2 UDB's *SELECT INTO* statement. SQL Server's *SELECT INTO* statement is equivalent to a *CREATE TABLE* statement followed by an *INSERT* statement in DB2 UDB. Thus, the following query in SQL Server:

```
SELECT * INTO t2 FROM t1
```

is equivalent to the following statements in DB2 UDB:

```
CREATE TABLE t2 AS (SELECT t1.* FROM
```

28.2.8. ANSI joins

Similar to VFP and SQL Server, the DB2 UDB syntax for joins is ANSI-style, with the operators:
_ *INNER*
_ *LEFT [OUTER]*
_ *RIGHT [OUTER]*
_ *FULL [OUTER]*
VFP and SQL Server display all *NULL* values at the beginning of the result set when the *ORDER BY* or *GROUP BY* clause is included in a *SELECT* statement. In DB2 UDB, *NULL* values of a column appear last when that column is ordered in ascending sequence, and first when ordered in descending sequence. Although DB2 UDB does not provide syntax for changing the sort order of *NULL* values, there is a way to obtain similar results by using the *COALESCE()* function to convert *NULL* to an empty string.

28.2.9. Autoincrement

The SQL Server *IDENTITY* property can be specified in the table definition to provide system-generated values in sequence. DB2 UDB also provides an *IDENTITY* attribute that can be specified in the table definition. The syntax used in DB2 UDB differs from SQL Server and VFP.

VFP	SQL Server	DB2 UDB
CREATE TABLE employee (; empid **INT AUTOINC**,; name VARCHAR(40) NOT NULL,; job VARCHAR(15) NOT NULL,; hire_date DATETIME NOT NULL,; department INT NULL,; basic_salary DECIMAL(8,2)	CREATE TABLE [employee] ([empid] **INT IDENTITY**, [name] VARCHAR(40) NOT NULL, [job] VARCHAR(15) NOT NULL, [hire_date] DATETIME NOT NULL, [department] INT NULL, [basic salary] DECIMAL(8,2)	CREATE TABLE employee (empid **INT GENERATED ALWAYS AS IDENTITY**, name VARCHAR(40) NOT NULL, job VARCHAR(15) NOT NULL, hire_date TIMESTAMP NOT NULL, department INT,

NULL,; commission DECIMAL(8,2) NULL)	NULL, [commission] DECIMAL(8,2) NULL)	basic_salary DECIMAL(8,2), commission DECIMAL(8,2))

DB2 UDB only allows a single column in a table to be defined as *IDENTITY*. If unique sequences are required for additional columns, sequence objects can be used. Unlike *IDENTITY* columns, sequence objects are separate database objects and are not defined in the table definition.
The *IDENTITY_VAL_LOCAL()* function can be used to retrieve the last generated identity value. Note that this function returns the last value used in the same unit of work.

28.2.10. Referential constraints

SQL Server referential constraints have two possible actions: *CASCADE* and *NO ACTION* (default).
VFP referential constraints have three possible actions: *CASCADE*, *RESTRICT* and *IGNORE* (default)
DB2 UDB referential constraint definitions have four possible actions: *NO ACTION* (the default), *RESTRICT*, *CASCADE*, and *SET NULL*.
When converting to DB2 UDB, no change in syntax is required.

28.3. Indexes

Visual FoxPro supports the following types of index files: structural compound index (.cdx), nonstructural compound index (.cdx) files, and standalone index (.idx) files. The structural and nonstructural .cdx files can contain multiple indexes, which have names, or tags, that identify them, while the standalone .idx file contains only a single index.
SQL Server uses a B-tree data structure to store clustered and non-clustered indexes. However, non-clustered indexes are not ordered physically according to the index keys and the leaf nodes consist of index rows rather than data pages.
DB2 UDB creates indexes in a separate data structure that replicates the keys' values. A B+ tree data structure is used to store indexes. To maintain the cluster factor of a clustered index or improve it dynamically as data is inserted into the associated table, DB2 UDB attempts to insert new rows physically close to the rows with index key values in the same range.
In both SQL Server and DB2 UDB, only one clustered index per table is permitted. In a DB2 UDB non-clustered index, the *NONCLUSTERED* clause is assumed by default.
There is a small difference in syntax used to create the indexes. In DB2 UDB, the *CLUSTERED* clause proceeds the index definition.

VFP	**SQL Server**	**DB2 UDB**
INDEX ON author_id TAG author_id ASCENDING	CREATE CLUSTERED INDEX [PK_author_id] ON [authors] ([author_id] ASC)	CREATE INDEX PK_author_id ON authors (author_id ASC) CLUSTER

VFP indexed may be created based on any valid VFP expression.
Both in SQL Server and DB2 UDB indexes cannot include expressions. It is possible to create an index on a single column or composite index on list of columns.

29. Data access

29.1. ActiveX Data Object (ADO)

Generally, if the OLE DB or ODBC interfaces are used, fewer application code changes are required than if a native driver is used. Converting from Microsoft OLE DB to IBM OLE DB requires some change, but converting from Microsoft ODBC to IBM ODBC is relatively straightforward and few changes are required.

29.2. Establishing connection

The IBM OLE DB Provider for DB2 allows DB2 UDB to act as a resource manager for the OLE DB provider. Example 10-5 shows a comparison of a typical connection string between the SQL Server OLE DB driver and the DB2 UDB OLE DB driver.

29.2.1. Example for OLE DB connection strings

Microsoft OLE DB for SQL Server:

```
“Provider=SQLOLEDB;Data Source=myServerName;Initial Catalog=
myDbName;User ID= myUserName;Password= myPwd;”
```

IBM OLE DB for DB2 UDB:

```
“Provider=IBMDADB2;Data Source=REDBOOK;UID=userid;PWD=password;”
```

29.2.2. Example for ODBC connection strings

SQL Server:

```
DRIVER={SQL Server};SERVER= myServerName; uid=myUserName;
pwd=myPwd;DATABASE= myDbName;
```

IBM DB2 ODBC

```
“driver={IBM DB2 ODBC DRIVER}; Database=myDbName; hostname=myServerName;
port=myPortNum;protocol=TCPIP; uid=myUserName; pwd=myPwd”
```

No significant differences in:
Views
Defaults

30. Differences not considered in this document:

Strong type casting
Variable assignments
String considerations
Case sensitivity
Unspecified FROM clause
Cross join
TOP n PERCENT
Cursors
Built-in SQL functions except mentioned above
Date arithmetic:

    DATEADD(day, 1, GETDATE())-> CURRENT DATE + 1 day
    DATEADD(month, 1, GETDATE()) -> CURRENT DATE + 1 month
    DATEADD(year, 1, GETDATE()) -> CURRENT DATE + 1 year

XML processing
Temporary tables
Stored procedures
User defined functions
Triggers
Transaction logging
Administrative issues

31. DB/2 Support

```
********************************************
* Tests run with MS SQL Server 2005 and     *
* IBM UDB DB2 installed on the same machine *
********************************************
```

31.1.1. Step 1: Install DB2

DB2 UDB Express edition

Custom Installation
    Select all available features
Next

Check 'Install DB2 Universal Database Express Edition on this computer' checkbox
Next

Install path:
    C:\Program Files\IBM\SQLLIB\ (default)
Selected languages:
    English
Next

DB2 Information Center:
    Select the location from which you will access the DB2 Information Center:
    Select 'On IBM Website' option
Next

    U: sa-ibm
    P: SA_0123456789
    Check "Use same user and password for remaining DB2 services"
Next

Administration contact list location
    Select 'Create a contact list on this system' option
Next

DB2 Instances:
    DB2
        Name: db2c_DB2
        Port: 50000
        (Default)
Next

Select the metadata you want to prepare
    Select 'Prepare the warehouse control database' option
Next
Next

When a health monitor threshold is breached an email or pager notification will be sent to the administrator
Specify a contact for health monitor notification:
    Select "Defer the task until after installation ..." option
Next

Used when synchronizing DB2 servers with a DB2 Control Server
Request satellite information
    Check "Defer the task until after installation ..." checkbox

Next...

Enable operating system security for DB2 objects
Check ‘Enable operating system security’ checkbox
Next

Install

==>

Setup is complete

****

31.1.2. Step 2:

Upsize GoMa DBC to MS SQL 2005 (through VFX) 'goma'
!!! WHEN UPSIZING MAKE SURE NO TIMESTAMP COLUMNS ARE ADDED !!!

Create ODBC DSN for SQL2005 server
(in ControlPanel->AdministrativeTools->DataSources)
This DSN is used later by IBM MTK to migrate SQL DB to DB/2

31.1.3. Step 3:

Using the IBM MTK(migration tool kit v 1.4.5, NOT wizard) to transfer a DB from SQL2005 to DB2 Express Edition
http://www-306.ibm.com/software/data/db2/migration/mtk/
1.Project management dialog
Proj. name: <enter MTK project name>
Source db type: Microsoft SQL Server
Target db type: DB2 UDB 8.2 for linux, UNIX, Windows

2.'Specify source' page
Press *Extract ...*
JDBC/ODBC DSN alias: <enter name of ODBC DSN for SQL2005 server>
UserID: <userid for the SQL2005 server>
Password:<password for the SQL2005 server>
Press *OK*
Extract dialog
Check the DB from SQL2005 to be extracted ('goma')
<Enter script file name>
Press *Extract*

3.'Convert' page
Press *Convert*

4.'Generate Data Transfer Scripts' page
Press *Generate Scripts*

5.'Deploy to target' page
DB2 database name: <enter name for the db in DB2, less or equal to 8 chars> 'goma_db2'
Select 'Use a local database' option and mark '(Re)create' checkbox
Enter User ID and Password for the Db2 server (sa-ibm/SA_0123456789)

Mark 'Extract and store data on this system' checkbox
Mark 'Load data to target database using generated scripts' checkbox
Press *Deploy*

And here is the connection string to connect to DB/2 database
DRIVER={IBM DB2 ODBC DRIVER};
UID=sa-ibm;PWD=SA_0123456789;DBALIAS=GOMA_DB2;

32. COM Server

COM Server is a server component, designated to work as web service, executing select commands or command scripts. Along with the command (or script) to be executed, the Execute method receives as parameters, domain, username and password.
Das COM Server-Objekt wird ohne Parameter instanziiert.

32.1. Die COM Server Klasse

32.1.1. Methoden

Execute *(tcSelectCmd, tlScript, tnResultType, tlReturnErrorArray, tcResultObjectName, tcDataXML, tcPath, tlTransaction, tcUserName, tcPassword, tcDomainName)* – executes a select query or script and returns the result as an XML string, Array or Variable.

Parameter

tcSelectCmd	Zeichenkette mit dem Select Befehl oder dem auszuführenden Skript.
tlScript	when =.T., the passed *tcSelectCmd* will be run isung ExecScript() function, when this parameter is .F., the content of *tcSelectCmd* is considered to be a single command;
tnResultType	Typ des Rückgabewertes: 0 or .F. – XML Zeichenkette 1 – Array 2 – Variable
tlReturnErrorArray	when this parameter is .T., in case of error the method Execute returns an error array instead of expected result;
tcResultObjectName	Name des Ergebnisobjekts (Cursor, Array oder Variable)
tcDataXML	local cursors, sent to the service. The content of this string is transformed to cursors before excuting command or script and is available to be used in them. It is possible to include several cursor in this XML string;
tcPath	Zusätzliche Pfadangabe, falls erforderlich.
tlTransaction	if this parameter is .T. and also the parameter *tlScript* is .T., the script will be executed into transaction. In case of error, the transaction is rolled back;
tcUserName	Benutzername für die Impersonate Anmeldung.
tcPassword	Kennwort für die Impersonate Anmeldung.
tcDomainName	Name der Domain für die Impersonate Anmeldung.

Rückgabewert

XML string, array or variable, containing the resultant cursor. An empty string is returned in case of error, if *tlReturnErrorArray* parameter is .F.. An error may occurred but during impersonation or during the execution. If an error occurred during the execution, the error information is logged in a text file *ErrorLog.txt* in current folder. Note that is the component works as web service, current folder is *Windows\System32*. If *tlReturnErrorArray* is .T., an array created with AERROR() function will be returned.

Bemerkungen

- When tlScript is .T., the parameter *tcResultObjectName* is required. If a single command is passed for execution and *tcResultObjectName* is empty, a temporary cursor name is generated and INTO CURSOR clause is appended to the eceuted command

- When tcDataXML is passed, it is required to have MSXML 4.0 installed on the computer. If MSXML 4.0 is not installed, an error is logged and the execution is cancelled. In such case an empty string is returned.

Wichtig
Ein Array wird als Rückgabewert von einem Web Service nicht unterstützt. Wenn der COM Server als Web Service eingesetzt wird, ist es nicht möglich als Rückgabewert ein Array zu liefern.

Die COM Server-Klasse besitzt drei versteckte Methoden: Impersonate, CheckRequiredComponents und LogError, die hier nicht weiter erläutert werden sollen.

32.2. Sicherheitsaspekte

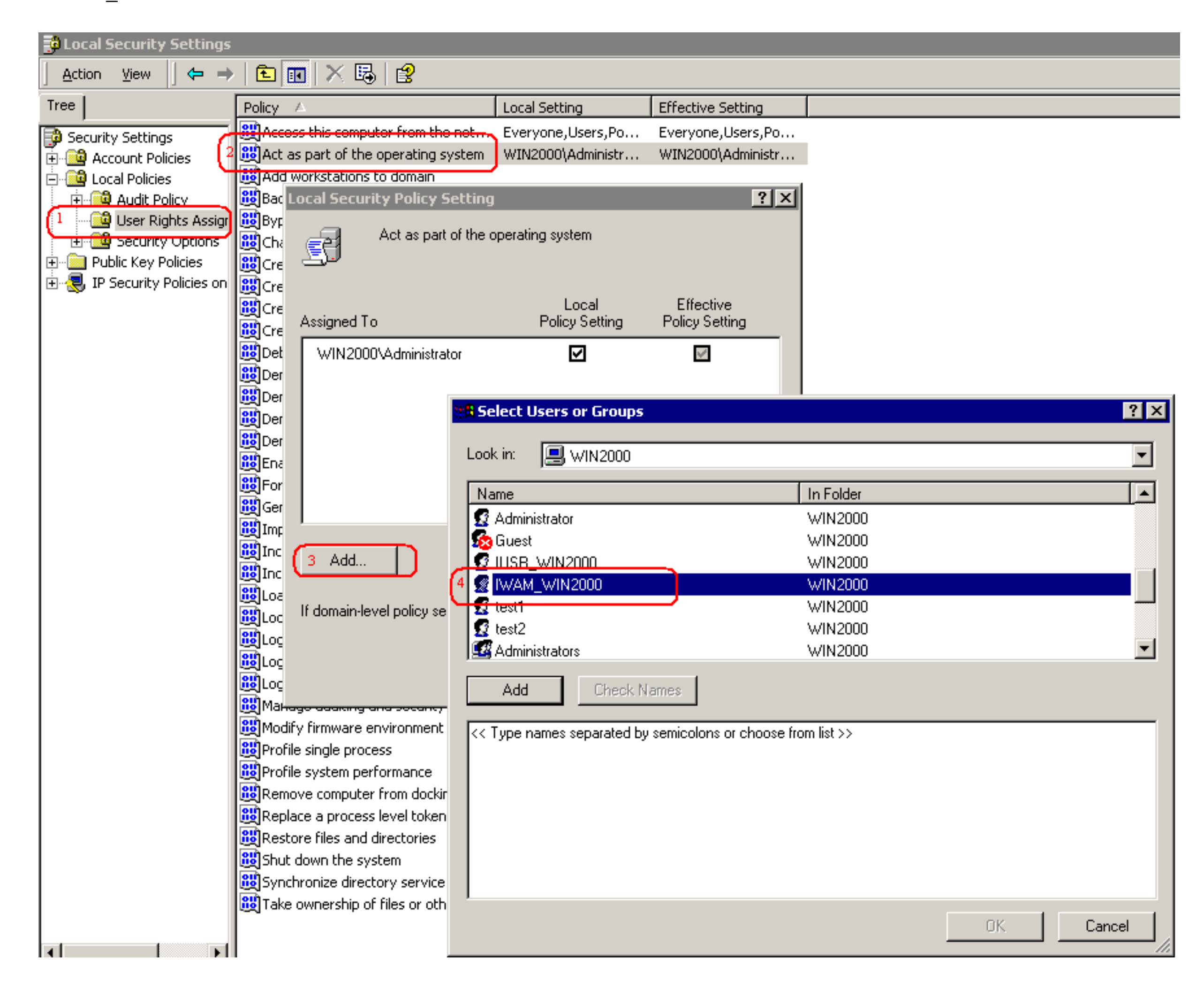
32.2.1. Skriptausführung

When ExecScript is executed, VFP writes the script as temporary program file. This temporary file is created into Temp folder and thus the user account, under which the script is executed, must have appropriate rights on that folder.
This can be achieved in two different ways: either administrator can assign necessary rights to the user account or a config.fpw is created for the component, redirecting temporary folder to a folder, whara the user account is allowed to write files. In the COMServer project file is included a config file for this purpose.

32.2.2. Impersonation

Die Methode Impersonate benutzt die API Funktion LogonUser, die spezielle Privilegien des Prozesses erfordert, der diese Methode aufruft.
Für Windows 2000: Der Prozess, der LogonUser aufruft, muss das Privileg SE_TCB_NAME besitzen. Wenn der aufrufende Prozess dieses Privileg nicht besitzt, führt die Ausführung von LogonUser zu einem Fehler und GetLastError liefert den Rückgabewert ERROR_PRIVILEGE_NOT_HELD. In einigen Fällen muss der Prozess, der LogonUser aufruft auch das Privileg SE_CHANGE_NOTIFY_NAME besitzen, sonst führt die Ausführung von LogonUser zu einem Fehler und GetLastError liefert den Rückgabewert ERROR_ACCESS_DENIED. Dieses Privileg ist nicht erforderlich für das lokale Systemkonto sowie für Konten, die Mitglied der Gruppe Administratoren sind. Standardmäßig ist das Privileg SE_CHANGE_NOTIFY_NAME für alle Benutzer aktiviert, es kann aber von Administratoren deaktivert werden. Weitere Informationen über Privilegien können hier nachgelesen werden: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/secauthz/security/privileges.asp .
Das Privileg SE_CHANGE_NOTIFY_NAME kann aktivert werden, in dem man dem Benutzer *Security Settings\Local Policies\User Rights Assignment\Act as part of the operating system* einträgt, siehe auch http://www.derkeiler.com/Newsgroups/microsoft.public.platformsdk.security/2004-06/0106.html .

Wenn die COM Server DLL als Web Service eingesetzt wird, läuft der Prozess mit den Rechten des Benutzerkontos IWAM.
See the screenshot below for details (the name of test computer is WIN2000, so IWAM account is IWAM_WIN2000:



32.3. Test code

In TestProc subfolder is places TestCOMServer.RPG file contaning code used to test COM Server both as a COM component and as web service.

32.4. Enhancement ideas

To create a separate method to impersonate the process, so that it is possible to call Execute method several subsequent times without passing user data to it. This will be more convenient if several queries are executed one after another using one and same instantiated object.

33. Vorbereiten einer Anwendung für die Produktaktivierung

33.1. Einstellungen im VFX – Application Builder

Im VFX – Application Builder muss zunächst die Produktaktivierung eingeschaltet werden. Dies geschieht mit der Checkbox „Enable Product Activation“. Die Wert der Eigenschaft cFoxAppl.lUseActivation kann wahlweise im Klassen-Designer auch manuell auf .T. eingestellt werden.

Der Aktivierungsschlüssel wird in einer Datei gespeichert. Der Name dieser Datei kann im VFX – Application Builder unter „Store activation data to“ eingetragen werden. Manuell kann der Wert der Eigenschaft cVFXActivation.cStoreActivationData eingestellt werden. Der Standardwert ist VFX.ini.

Es besteht die Möglichkeit vom Web Service oder über die HTTP Aktivierung Aktivierungsschlüssel automatisch erstellen zu lassen, die zeitlich befristet sind. So kann man interessierten Kunden die Möglichkeit geben, die Anwendung in einem festgelegten Umfang testen zu können. Hier muss im VFX – Application Builder „Activation key validity in days“ eingestellt werden. Der Standardwert ist 30 Tage.

Bei „Activation key type“ kann das zu bisherigen VFX Versionen kompatible Format des Aktivierungsschlüssels gewählt werden. Diese Aktivierungsschlüssel können relativ lang werden. Wahlweise kann auch ein kürzeres Format für den Aktivierungsschlüssel verwendet werden. Bei diesem Format ist der Aktivierungsschlüssel immer genau 25 Zeichen lang. Jeweils fünf Stellen sind durch einen Bindestrich getrennt. Anwender kennen dieses Format von Aktivierungsschlüsseln von verschiedenen Microsoft Produkten. Im VFX – Application Builder wird „Activation key type“ eingestellt. Manuell kann der Wert der Eigenschaft cVFXActivation.nProductActivationBehavior auf 1 für lange Aktivierungsschlüssel oder 2 für kurze Aktivierungsschlüssel eingestellt werden.

Die Checkbox „Time limited activation key“ muss markiert werden, wenn befristet gültige Aktivierungsschlüssel erstellt werden sollen. Diese Checkbox ist nur dann enabled, wenn kurze Aktivierungsschlüssel verwendet werden. Manuell kann der Wert der Eigenschaft cVFXActivation.lUseTimeLimitedActivationKey eingestellt werden.

Im Eingabefeld „Start day of activation key“ kann das Startdatum eingegeben werden, ab dem befristet gültige Aktivierungsschlüssel erstellt werden können. Der Standardwert ist der 01.01.2007. Manuell kann der Wert der Eigenschaft cVFXActivation.dStartActivationDate eingestellt werden.

*** Noch nicht fertig im App Builder

Appl.vcx – cvfxactivation
cHTTPRegisterURLObjectName = „Register.asp“
cHTTPRegisterURLServerName = dla.homeip.net (ohne http:// und ohne / am Ende!
www.outsourcingITservices.net/RegisterTest

cstorehardwareparameters=“vfx.hrd“ nur bei hardware toleranz?

cRegEMail = „uweh@prolib.de“
nHardwareParametersTolerance = 1
nRegWay = 13
cRegisterFormName = „vfxactivationwizard“

33.2. Weitere manuelle Einstellungen

Es ist empfehlenwert zusätzlich in der Klassenbibliothek Appl.vcx in der Klasse cfoxappl die Werte der Eigenschaften cfoxappl.cappname und cfoxappl.ccompanyname einzutragen. In der Eigenschaft cappname sollte der Name der Anwendung eingetragen werden. In der Eigenschaft ccompanyname sollte der Firmenname eingetragen werden.

Wenn beide Eigenschaften mit Werten gefüllt sind, wird entsprechend der Werte auf dem Kundenrechner ein Ordner angelegt. Wenn der angemeldete Benutzer Schreibrechte im Ordner

C:\Dokumente und Einstellungen\Alle Benutzer\Anwendungsdaten

hat, wird in diesem Ordner die Datei mit dem Aktivierungsschlüssel gespeichert. Wenn der angemeldete Benutzer in dieem Ordner keine Schreibrechte hat, wird die Datei mit dem Aktivierungsschlüssel im Ordner

C:\Dokumente und Einstellungen\<Anmeldename des angemeldeten Benutzers>\Anwendungsdaten

gespeichert. In diesem Fall ist die Aktivierung nur für den angemeldeten Benutzer gültig.

Wenn die Werte der Eigenschaften cfoxappl.cappname und cfoxappl.ccompanyname leer sind, wird die Datei mit dem Aktivierungsschlüssel im Ordner der Anwendung gespeichert.

33.3. Einstellungen in VFX – Define Activation Rules

Im Assistenten VFX – Define Activation Rules werden die Hardware-Parameter ausgewählt, die für die Produktaktivierung verwendet werden. Hier muss mindestens ein Wert ausgewählt werden, aber auch eine beliebige Kombination aus Werten ist möglich.

Auf der Seite „Rights“ werden Berechtigungen eingetragen. Die eingegebenen Namen müssen gültige Namen für Eigenschaften sein. Zur Lafzeit der Anwendung können die Werte der Eigenschaften im Objekt goprogram.securityrights geprüft werden.

33.4. Build register DLL

Jetzt kann der COM Server erstellt werden, mit dessen Hilfe die Aktivierungsschlüssel erstellt werden. Der COM Server muss in der VFX – Kundenverwaltung zur Verfügung stehen. Wenn die Aktivierung auf einem Webserver durchgeführt wird, muss der COM Server auch auf dem Webserver registriert werden. Der COM Server kann als Web Service verwendet werden.

VFX stellt das fertige Projekt für den COM Server im Ordner RegisterDLL unterhalb des Projektordners zur Verfügung. Alle im COM Server benötigten Einstellungen werden von den VFX Buildern gemacht. Manuelle Änderungen oder Erweiterungen am COM Server sind in der Regel nicht erforderlich, aber natürlich möglich.

Über den Menüpunkt „Build register DLL“ kann der COM Server automatisch generiert und auf dem Entwicklungsrechner registriert werden. Der COM Server wird dabei als multithreaded DLL erstellt. Dies ist unbedingt zu beachten, wenn der COM Server manuell erstellt werden soll.

Wenn auf dem Entwicklungsrechner Windows Vista läuft, muss VFP explizit mit Administratorrechten gestartet werden, damit ausreichend Rechte vorhanden sind um den COM Server erstellen und registrieren zu können.

33.5. Einstellungen in der VFX – Kundenverwaltung

Die VFX – Kundenverwaltung unterstützt jeden Datenzugriff, der mit allen VFX Anwendungen möglich ist. Mit Manage Config.vfx wird die zu verwendende Datenbank mit den Kundendaten der Anwendung

muss der Klassenname in Regdllname eingetragen werden.

VFX.fll, Vfxlog,* und Config.vfx aus der Kundenverwaltung müssen in den Ordner der DLL! Bz. Der COM Server muss im Ordner der Kundenverwaltung laufen.

Bei W2003 muss IUSR Vollzugriff auf den Datenordner mit den Kundendaten bzw. Registrierungsdaten haben.

in WS dll (which also is use by asp but not as ws) is made a new methos...

CallMethodByName(tcMethodName as String, tcData as String) as String
all calls to asp page must pass method name and data string and asp simply calls this method of the registration object
in this way it is possible to add new methods without need of changes in asp
yes, but it is very easy to change. and yes... when you have 1 config with all the applicatiomns supported, it is not necessary to have more applications
cconnectwebservice.registerviahttp

33.6. Einstellungen im Internet Information Server 7

Die ASP Seite für die HTTP Registrierung kann auf mit dem Internet Information Server 6 auf Windows 2000, Windows XP oder Windows Server 2003 ohne besondere Einstellungen ausgeführt werden.

Für den Internet Information Server 7, der standardmäßig auf Windows Vista installiert wird bzw. auf älteren Windows Versionen nachträglich installiert werden kann, müssen einige Einstellungen vorgenommern werden.

Wenn der IIS auf einer 64 bit Windows Version läuft, muss der IIS so eingestellt werden, dass er im 32 bit Modus läuft. Nur so können VFP Objekte instanziiert werden. Das geschieht mit dem Befehl:

```
cscript %SYSTEMDRIVE%\inetpub\adminscripts\adsutil.vbs SET
W3SVC/AppPools/Enable32bitAppOnWin64 1
```

Als Rückmeldung sollte erscheinen:

```
Enable32bitAppOnWin64          : (BOOLEAN) Wahr ??
```

Danach muss IIS neu gestartet werden.

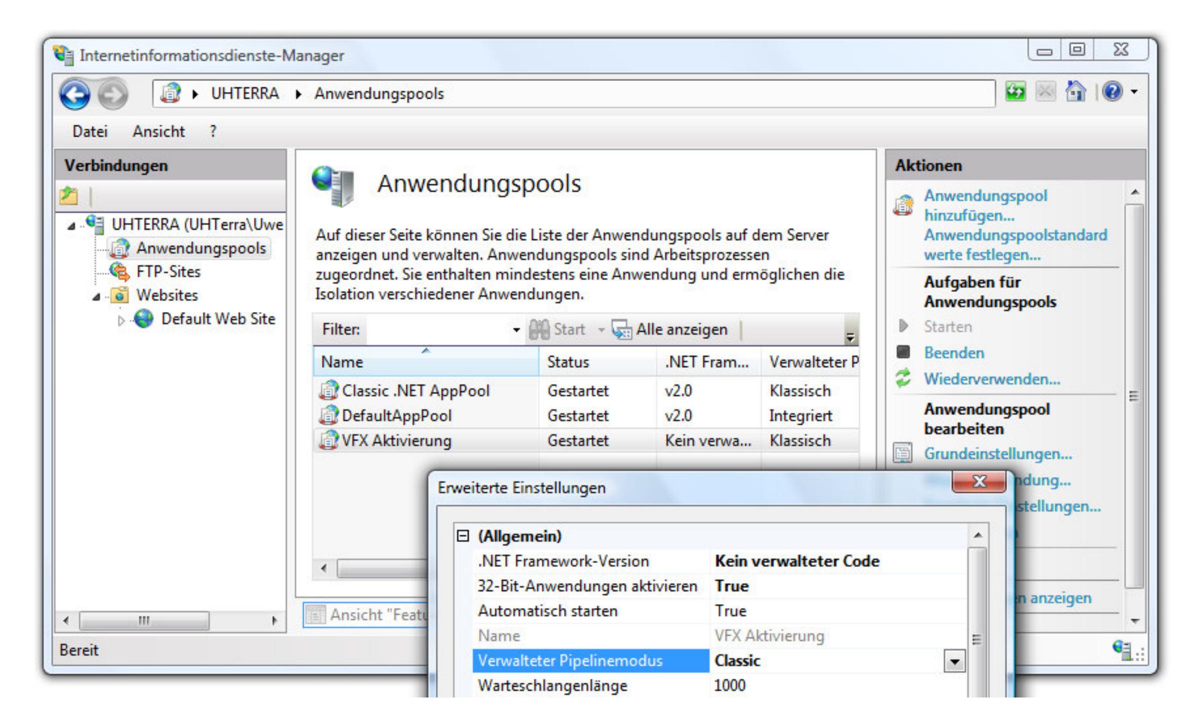
Im Internetinformationsdienste-Manager muss bei Anwendungspools ein neuer Anwendungspool hinzugefügt werden. Der Name kann beliebig gewählt werden, zum Beispiel VFX-Aktivierung. Folgende Einstellungen sind für den neuen Anwendungspool zu machen:

.NET Framework Version	Kein verwalteter Code
Verwalteter Pipeline Modus	Klassisch

Anschließend ist in den erweiterten Einstellungen des neu erstellten Anwendungspools

32-bit-Anwendungen aktivieren	True

einzustellen.



Unter Websites ist unter der verwendeten Website, in der Regel „Default Web Site", eine Anwendung hinzuzufügen. Dafür ist der Anwendungspool auf den im letzten Schritt erstellten Anwedungspool einzustellen, zum Beispiel VFX-Aktivierung.

Auf der Seite ASP im Internetinformationsdienste-Manager muss bei COM Eigenschaften

In MTA ausführen	True

eingestellt werden.

